

ROUTERARENA: AN OPEN PLATFORM FOR COMPREHENSIVE COMPARISON OF LLM ROUTERS

Anonymous authors

Paper under double-blind review

ABSTRACT

Today’s LLM ecosystem comprises a wide spectrum of models that differ in size, capability, and cost. No single model is optimal for all scenarios; hence, LLM routers have become essential for selecting the most appropriate model under varying circumstances. However, the rapid emergence of various routers makes choosing the right one increasingly challenging. To address this problem, we need comprehensive router comparison and a standardized leaderboard, similar to those available for models. In this work, we introduce ROUTERARENA, the first open platform enabling *comprehensive* comparison of LLM routers. ROUTERARENA has (1) a principally constructed dataset with broad knowledge domain coverage, (2) distinguishable difficulty levels for each domain, (3) an extensive list of evaluation metrics, and (4) an automated framework for leaderboard updates. Leveraging our framework, we have produced the initial leaderboard with detailed metrics comparison as shown in Figure 1. We will make our platform open to the public; the current code base is available here: <https://anonymous.4open.science/r/RouterArena>.

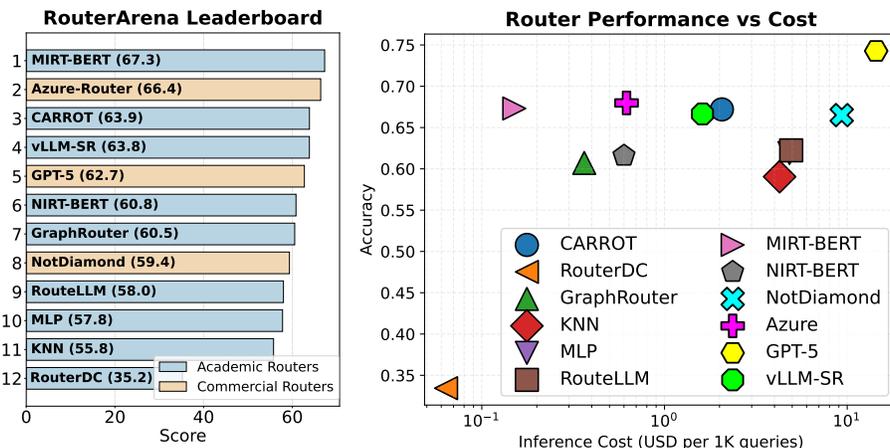


Figure 1: A quick view of ROUTERARENA leaderboard and performance-cost trade-off.

1 INTRODUCTION

Large Language Models (LLMs) are rapidly diversifying, offering an ever-wider spectrum of capabilities and inference costs. This diversity increasingly challenges the prevailing LLM usage pattern in which users manually choose models for their queries. The difficulty stems from the fact that no single model is universally optimal: powerful models excel at complex tasks but are costly, while smaller models are more efficient yet may struggle on difficult queries. As a result, *LLM routers* that automatically select models based on input queries are increasingly recognized as a core system primitive in practical deployments.

Given its importance and promise, many LLM routers have recently emerged in both industry and academia (Figure 2). A notable example is GPT-5 (OpenAI, 2025), which incorporates routing as a key feature by directing user queries to the most suitable model within the OpenAI family. As

054 routers proliferate, the challenge shifts from selecting the right model to selecting the right router.
 055 Unfortunately, **router evaluation has not kept pace: there is currently no open evaluation plat-**
 056 **form**, akin to LMArena (Chiang et al., 2024), that systematically compares open-source routers (Hu
 057 et al., 2024; Zhuang et al., 2024a) and commercial routing services (NotDiamond, 2025; Microsoft,
 058 2025) under a unified protocol.

059 It is urgent to fill this gap by building a *Router Arena* that can comprehensively evaluate and rank
 060 routers, enabling users to understand the status quo and make informed choices. However, unlike
 061 model arenas, designing a router arena is considerably more challenging due to the requirements
 062 from three key aspects. (1) **Dataset**. To evaluate whether a router can recognize problem domains
 063 and dispatch queries to appropriate models at minimal cost, the arena dataset must cover a broad
 064 range of domains and subjects, as well as varying difficulty levels. (2) **Metrics**. Router performance
 065 is inherently multi-dimensional, and so should be the arena ranking. While accuracy and cost are
 066 the primary metrics, it is also important to capture other dimensions such as routing optimality and
 067 robustness. (3) **Framework**. To enable live leaderboard updates, the arena must have a user-friendly
 068 framework that can automatically evaluate new open-source and commercial routers. Although ex-
 069 isting studies explore some of these directions, as summarized in Table 1 and discussed in Section 2,
 070 they fail to address each challenge in a *comprehensive* way.

071 In this work, we present ROUTERARENA, the first open platform for comprehensive evaluation and
 072 comparison of LLM routers. It addresses the above key challenges with the following designs:

- 073 • **A Principled Diverse Dataset**. To ensure broad coverage, we construct the dataset using the
 074 Dewey Decimal Classification system adopted in libraries, covering all domains except religion.
 075 For each subject, we apply Bloom’s taxonomy to design queries at three difficulty levels, produc-
 076 ing a diverse dataset of $\sim 8,000$ queries spanning 9 domains and 44 categories for router evaluation.
- 077 • **Extensive Metrics for Arena Ranking**. We construct router leaderboards by considering an ex-
 078 tensive list of deployment-relevant metrics including query-answer accuracy, query-answer cost,
 079 routing optimality (cheapest correct selection), robustness to query perturbations (consistency),
 080 and router overhead (latency). This enables router comparison from multiple perspectives.
- 081 • **An Automated Framework for Leaderboard Updates**. We design a framework that automat-
 082 ically evaluates new routers, collects metrics, and updates the leaderboard. The framework sup-
 083 ports both open-source and commercial routers, and employs prefix caching to improve efficiency.

084 Figure 1 provides a quick view of our accuracy–cost leaderboard along with other details. We have
 085 found that although GPT-5 achieves higher accuracy, its cost is significantly higher than that of other
 086 routers due to its model pool being restricted to the OpenAI family. Consequently, it does not rank
 087 as the best router on our accuracy–cost leaderboard.

088 Our vision is for ROUTERARENA to serve as an open community venue for evaluating routers as
 089 the ecosystem evolves, providing a standardized basis for fair comparison and progress tracking. By
 090 lowering the barrier to evaluation and enabling transparent, reproducible results, ROUTERARENA
 091 will help researchers and practitioners design, improve, and adopt better routers.

093 2 MOTIVATION

095 **The Rapid Emergence of LLM Routers**. As shown in Figure 2, the landscape of LLM routers
 096 is rapidly expanding, evolving from academic exploration to commercial deployment. From a few
 097 scattered academia routers (Zhang et al., 2023; Chen et al., 2023; Hari & Thomson, 2023) in mid-
 098 2023, the number of publications expanded to more than a dozen by 2024 (Liu et al., 2024; Zhuang
 099 et al., 2024b; Chen et al., 2024a; Zhao et al., 2024). By 2025, not only did academia routers continue
 100 to grow (Wang et al., 2025; Huang et al., 2025a; Ding et al., 2025; Zhang et al., 2025b), but commer-
 101 cial products also emerged (NotDiamond, 2025; Microsoft, 2025), most notably GPT-5 (OpenAI,
 102 2025) with a built-in router.

104 **New Problem: How to Choose the Right Router?** Answering this question requires compre-
 105 hensive router comparisons to understand the current landscape. Such comprehensiveness entails
 106 dataset categories, difficulty levels, evaluation metrics, and router inclusion. However, our review
 107 of existing work reveals that **no such comprehensive comparisons are available today**. As shown
 in Table 1, the existing work falls short in the following aspects.

108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161

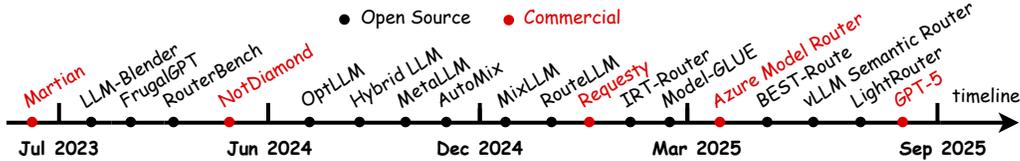


Figure 2: Timeline of example router-related works and products.

Table 1: Comparison of existing works (Hu et al., 2024; Huang et al., 2025b; Feng et al., 2025b; Zhuang et al., 2024a) and ROUTERARENA. ROUTERARENA enables comprehensive router comparison with extensive query categories, difficulty levels, evaluation metrics, and router inclusion.

Benchmark	Query categories	Difficulty levels	Evaluation Metrics	Commercial Routers	Router Ranking
RouterBench	24 Categories	✗ No analysis	✗ Deferral curve only	✗	✗
RouterEval	27 Categories	✗ No analysis	✗ Accuracy metric only	✗	✗
FusionBench	26 Categories	LLM-judge analysis	✗ Deferral curve only	✗	✗
EmbedLLM	26 Categories	✗ No analysis	✗ Accuracy metric only	✗	✗
ROUTERARENA	44 Categories based on DDC	✓ 5 Bloom Level Classification	✓ 5 Evaluation perspectives	✓ 3 Included	✓ Multi-metric leaderboard

- *Narrow Query Category Coverage.* They lack full coverage of query categories, making them impossible to evaluate router performance on queries from excluded categories.
- *Indistinguishable Difficulty Levels.* They do not differentiate queries by difficulty, limiting their ability to test accuracy–cost tradeoffs.
- *Incomplete Evaluation Metrics.* They only consider a subset of relevant metrics, overlooking important dimensions such as optimality, robustness, and latency.
- *No Support of Commercial Routers.* Current frameworks evaluate only open-source routers and do not extend to closed-source or commercial routers.
- *No Router Leaderboard.* There is no leaderboard that allows people to compare all routers under a unified evaluation protocol.

This Work: ROUTERARENA. This gap motivates us to design ROUTERARENA, an open platform for comprehensive router comparisons. In the remainder of this paper, we first introduce the key components of ROUTERARENA: principled dataset construction, comprehensive metric formulation, and an automated evaluation framework with live leaderboard. We then present our evaluation results and discuss the key findings.

3 ROUTERARENA EVALUATION DATASET

To enable meaningful and unbiased router comparisons, a high-quality evaluation dataset is essential. In this work, we construct such a dataset by adhering to three guiding principles.

Principle 1: DDC-Inspired Diverse Domain Coverage. To evaluate a router’s ability to recognize problem domains and route queries to the appropriate specialist models, the dataset must provide broad domain coverage. To achieve this, we draw inspiration from the Dewey Decimal Classification (DDC) system (Dewey, 1876), a book classification framework widely used in libraries. The DDC is renowned for its comprehensive and logical structure, providing a proven methodology for organizing the entire world of knowledge into distinct, hierarchical categories (Satija, 2013).

Principle 2: Bloom-Guided Cognitive Skill Levels. Routers in practice face queries that range from simple factual recall to complex reasoning. To ensure coverage of *cognitive skills* from simple recall to higher-order reasoning and judgment, we adopt Bloom’s taxonomy (Bloom et al., 1956), a widely used framework for quantifying question complexity and cognitive processes (Ullrich & Geierhos, 2021; Herrmann-Werner et al., 2024; Padó, 2017).

Principle 3: Empirically Defined Difficulty Levels. To test whether a router can make meaningful accuracy–cost tradeoffs—choosing between powerful but expensive models and weaker but cheaper ones—the dataset must also provide clearly distinguishable *difficulty* levels. We therefore define difficulty empirically: for each query, we compute the number of models (out of 42) that answer it correctly and sort queries by this empirical difficulty.

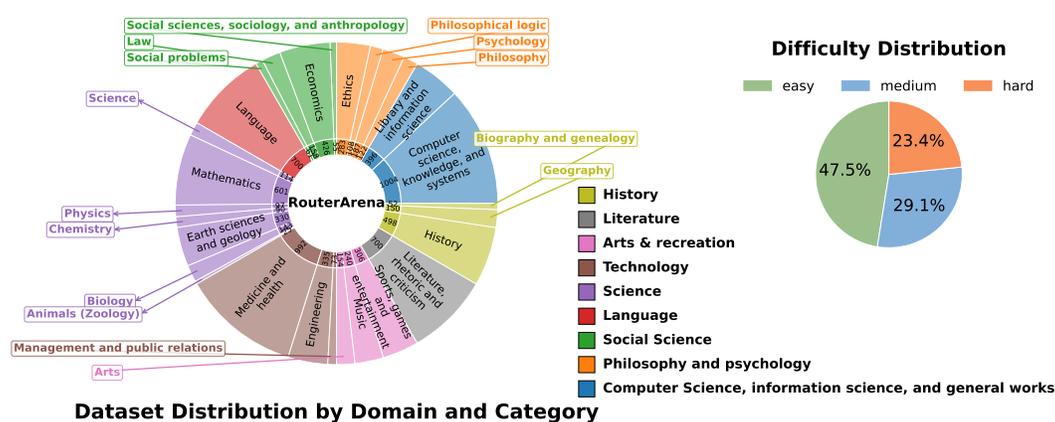


Figure 3: Dataset composition. For ease of demonstration, we merged some categories.

Dataset Construction Process. Following these two principles, we curate our evaluation dataset as follows. To ensure coverage across all DDC categories (excluding religion), we first collect all the queries from two existing LLM benchmark datasets and then supplement underrepresented categories with data from 21 open-source, domain-specific datasets, resulting in 62,000 raw queries. We then cap each source dataset at approximately 1,000 queries.¹ To determine the cognitive skill level of each query based on Bloom’s taxonomy, we employ an LLM-as-Judge approach with DEEPSEEK-V3.1 (DeepSeek-AI, 2025) (prompt specified in Appendix D). We use these Bloom levels only to characterize the cognitive skills involved in each query, not as ground-truth difficulty labels, and we further validate them with a small-scale human study (Appendix C.1).

Next, to fairly distribute questions across categories and cognitive levels, we propose a recursive deficit redistribution algorithm. We begin by setting the ratio of science to humanities at 2:1. Within each top-level category, if a sub-category falls short of its proportional quota, the resulting surplus is recursively and uniformly redistributed to those sub-categories that exceed their initial allocation. We apply the same procedure within each sub-category to allocate data across different cognitive levels, ensuring balanced coverage throughout the dataset.

However, this raw dataset contains many highly similar or even duplicate questions inherited from the various sampled sources. Such redundancy does not benefit router evaluation and may even introduce noise into the results. To address this, we perform cosine-similarity-based de-duplication using SENTENCE-TRANSFORMERS/ALL-MINI-LM-L6-V2. By strictly following the allocation strategy and selecting the least similar samples, we ensure that the resulting ROUTERARENA dataset maintains broad coverage with minimal redundancy.

The Resulting Dataset. Our final evaluation dataset consists of 8,400 queries sampled from 23 source datasets. It spans nine top-level domains and 44 categories in Figure 3 (Left). We define difficulty empirically: for each query, we count how many of the 42 models answer it correctly and group queries into three bands—hard ($\leq 4/42$), medium ($5-19/42$), and easy ($\geq 20/42$)—as shown in Figure 3 (right). We provide the model list of these 42 models and the rationale to use them at Appendix C.4. Figure 4 further shows that, when sorted by this measure, both the overall and per-domain curves increase smoothly from low to high accuracy without large gaps, indicating a well-spread spectrum of difficulty that can distinguish router behavior. Note that this empirical difficulty distribution is skewed, but it reflects real-world query patterns—easy and medium questions occurring more frequently than hard ones. We include more details about the dataset in Appendix C.2, including dataset schema and concrete query examples.

4 ROUTERARENA EVALUATION METRICS

ROUTERARENA supports comprehensive router evaluation along five dimensions.

¹It primarily affects large corpora such as WMT and SuperGLUE and yields roughly 30,000 candidate queries while preventing a few sources from dominating the pool

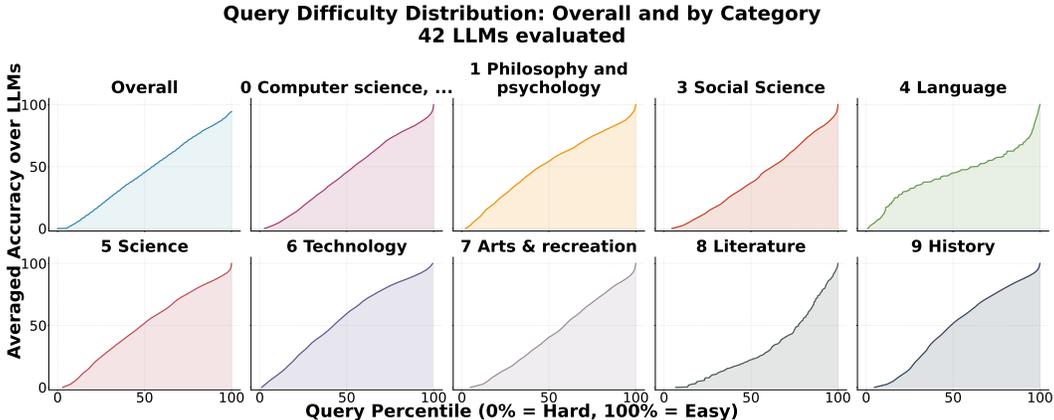


Figure 4: Each query is answered by 42 LLMs and evaluated for accuracy. Sorting queries by empirical accuracy yields smooth, monotonic curves overall and within each domain, indicating that query difficulty is well spread across the full range from very hard (left) to very easy (right).

(1) **Query-answer Accuracy.** This metric captures a router’s ability to direct queries to the appropriate models such that they are correctly answered. We calculate accuracy as the average correctness across all our dataset queries.

(2) **Query-answer Cost.** This measures the cost incurred by a router’s routing decisions. To address important factors such as the variable cost introduced by input length and generation length (e.g., in chain-of-thought reasoning) as well as the distinct computational characteristics of Mixture-of-Experts (MoE) models, we use the actual inference cost measured by:

$$cost = c_{in} * N_{in} + c_{out} * N_{out} \tag{1}$$

Where c is the cost per token and N is the number of tokens. We obtain the cost c for the specific models a router chooses using the official API pricing published by the corresponding providers (e.g., OpenAI, Claude, Fireworks AI, etc.). For unpopular models that are not served by commercial providers, we deploy them ourselves for experiments (only a few) and approximate their costs using the pricing tiers published by commercial hosting platforms (e.g., Together.ai), which estimate serving costs based on model size (parameter count) and architecture type (e.g., MoE) in Table 9.

(3) **Routing Optimality.** This captures a router’s ability to perform optimal routing—that is, selecting the cheapest model that still produces a correct response. It consists of three sub-metrics: (a) *Optimal Selection Ratio*—the proportion of queries for which the router answers correctly by selecting the cheapest model; (b) *Optimal Accuracy Ratio*—the ratio between a router’s achieved accuracy and the upper-bound accuracy obtainable when always selecting the best model from its model pool; (c) *Optimal Cost Ratio*—the ratio between the cost incurred by the router’s selections and the cost of always choosing the optimal model. This metric will penalize routers that rely on unnecessarily expensive models when cheaper, correct alternatives are available. This metric will penalize routers that use unnecessarily expensive models when cheaper, correct alternatives are available.

(4) **Routing Robustness.** This metric evaluates the router’s robustness against noisy inputs. We calculate it as the proportion of queries for which the router makes consistent routing decisions under perturbed input. Specifically, we generate noisy variants of queries—through paraphrasing, grammatical changes, synonym substitutions, and typos—and measure the percentage of cases where the router selects the same model as it does for the original, noise-free query. This captures the router’s capability for handling realistic, imperfect user queries.

(5) **Routing Latency.** Since the router operates in the critical path of systems in production, it must handle millions of queries per second with minimal overhead. This metric measures the additional latency introduced by routing. It reflects the latency increase in both time-to-first-token (TTFT) and end-to-end response latency when a given router is employed.

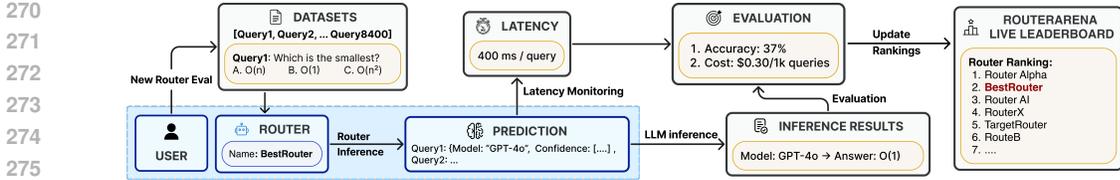


Figure 5: RouterArena Live Leaderboard.

5 ROUTERARENA EVALUATION FRAMEWORK

5.1 ARENA RANKING

ROUTERARENA provides a series of router leaderboards that enables users to compare the capabilities of different routers and select the one best suited to their scenarios. It includes six ranking scores based on the evaluation metrics described in Section 4, including Arena Score, Optimal-selection-ratio, Optimal-acc-ratio, Optimal-cost-ratio, Robustness, and Latency. Among these, the Arena score $S_{i,\beta}$ captures the trade-off between accuracy and cost by combining them into a single composite measure using the Weighted Harmonic Mean (Ferguson, 1931). Specifically, to better distinguish between routers with low costs, we apply a base-2 logarithmic (\log_2) transformation to the cost values. Under this scaling, each doubling of price reduces the cost score by one unit. For router i with cost c_i , we define its normalized cost as

$$C_i = \frac{\log_2(c_{max}) - \log_2 c_i}{\log_2(c_{max}) - \log_2(c_{min})} \quad (2)$$

where c_{max} and c_{min} denote the maximum and minimum costs of routing 1k queries. Specifically, we choose $c_{min} = 0.0044$, corresponding to the cost of the cheapest model in the leaderboard’s model pool. This reflects the cost of a router that always selects the cheapest model. We choose $c_{max} = 200$ representing the most expensive model, OpenAI’s O1-PRO. This normalization maps the cost into range $[0, 1]$, with larger values of C_i corresponding to more economical routers. Next, we combine the normalized cost C_i and accuracy A_i using a weighted harmonic mean:

$$S_{i,\beta} = \frac{(1 + \beta)A_i C_i}{\beta A_i + C_i}, \quad (3)$$

where the parameter $\beta > 0$ controls the relative importance of accuracy versus cost. Intuitively, β specifies the relative weighting applied to cost: we assign cost a weight of β in comparison to accuracy. By default, we use $\beta = 0.1$, emphasizing routing accuracy, because highly accurate routers are generally more valuable even if they incur slightly higher costs.

Details about how we update c_{min} and c_{max} , recommendations on values of β for different user needs are provided in Appendix H.

5.2 AUTOMATED EVALUATION FRAMEWORK

Although we demonstrate ROUTERARENA with a specific set of routers in this paper, it is very easy to update the leaderboard with new routers. To facilitate this process, we have designed an automated evaluation framework that will be released publicly alongside the leaderboard. Figure 5 shows the overall system workflow. To evaluate a new router, the user can simply start by providing our framework with an access point (e.g., an API) to the router. The framework sends evaluation queries to the router, which performs routing inference on its end and returns its model selections. To ensure fairness, we run the inference ourselves and use cached results when possible, since many routers share overlapping model pools. During this process, the router’s response time is monitored to measure routing latency. Finally, the framework computes the evaluation metrics and aggregates the results, which are reflected in the leaderboard. Note that some commercial routers may not expose their model selections and instead return query answers directly. Such routers can still be evaluated with our framework, although certain metrics cannot be measured in this setting.

6 EXPERIMENTS

6.1 EXPERIMENTAL SETTINGS

Router Selection. For commercial routers, we evaluated the router from Not Diamond (NotDiamond, 2025), which provides access to over 60 models, and the Azure Model Router (Microsoft, 2025), which currently only supports OpenAI models. We also included GPT-5 (OpenAI, 2025), whose model family incorporates an internal router. For NotDiamond, we selected 26 representative models spanning different parameter scales, architectures, and reasoning abilities. For Azure-Router, we evaluated the entire model pool, including GPT-5 model families. Appendix B provides full details of the model pools used for each router.

For open-source routers, we evaluated nine representative systems covering a diverse routing approaches. Specifically, we chose both the KNN- and MLP-based methods trained on RouterBench (Hu et al., 2024) as baselines. We further included GraphRouter (Feng et al., 2025a), which leverages graph neural networks (GNNs) for routing, and the Universal Router (Jitkrittum et al., 2025), which uses K-means clustering. To capture cost–accuracy tradeoffs, we evaluated CARROT Router (Somersstep et al., 2025), while RouterDC (Chen et al., 2024b) was incorporated as a dual contrastive learning–based approach. Additionally, we considered IRT-Router (Song et al., 2025), which applies item response theory to explicitly model the interaction between query attributes and model capabilities, and RouteLLM (Ong et al., 2025), which performs binary selection between a stronger and a weaker model. Moreover, we also take the latest vLLM Semantic Router (vLLM, 2025) into consideration, which leverages a ModernBERT (Hugging Face, 2025) to categorize the incoming requests into pre-defined categories, and selects the model that has the highest score.

Router Training and Evaluation. For commercial routers, no additional training is required; we simply accessed their provided APIs for evaluation. In contrast, for academia routers, we followed the training procedures and datasets specified in their open-source implementations. Specifically, we did not modify the training datasets or the task categorizations (if applicable). The model pools were configured in accordance with the original papers. In particular, for the vLLM-SR, we constructed the pool using both open-source models of varying parameter scales and proprietary models, with detailed configurations summarized in Table 3. After training each router, we evaluated them by feeding our benchmark dataset, recording the model selected, the latency incurred by the selection, and the confidence scores assigned to all candidate models in the pool.

6.2 RESULTS

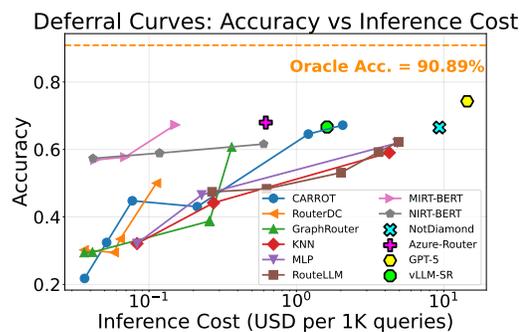


Figure 6: Deferral Curve: accuracy versus cost

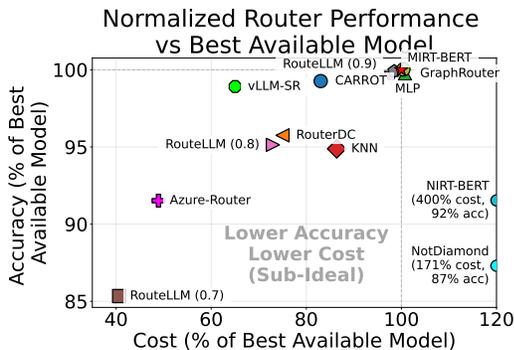


Figure 7: Normalized Scatter Plot

Deferral Curve. Figure 6 presents the trade-off between accuracy and inference cost. As we increase inference budget, we unlock more powerful models, driving the accuracy up. For open-source routers, we leveraged their confidence scores to apply budget-based masking, which produces multiple points along each curve. With only cheap models available, accuracy remains low, but as larger models enter the pool, routing accuracy increases. In contrast, commercial routers typically appear as single points because their model pools already include the best-performing models.

Two insights emerge. First, the orange dashed line shows the oracle accuracy, highlighting that all routers fall short of the best achievable performance. Second, the trade-off frontier differs by setting:

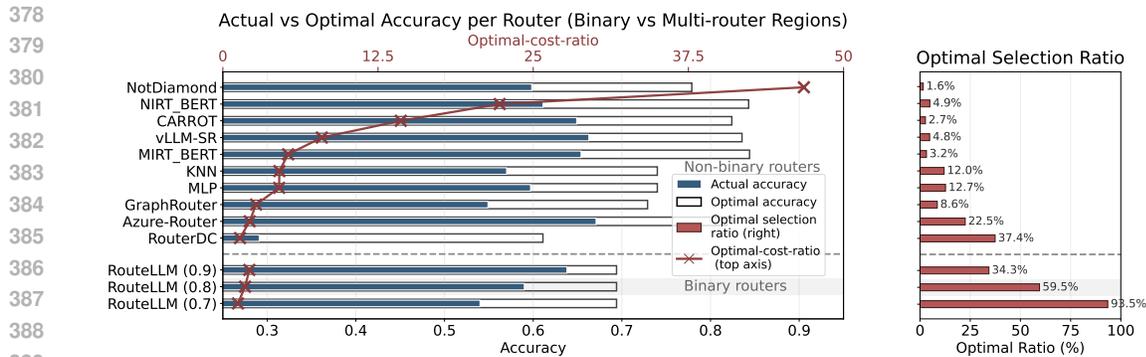


Figure 8: Actual and optimal accuracy, along with optimal selection ratio and cost ratio

commercial routers can achieve higher accuracy, but usually at significantly higher costs; open-source routers, on the other hand, achieve competitive performance at much lower budgets, though they plateau earlier. Notably, routers like CARROT and GraphRouter illustrate cost-efficient routing, while systems such as GPT-5 and NotDiamond lean heavily on expensive models for accuracy. This suggests that while commercial routers prioritize maximizing accuracy, academic approaches often explore the efficiency side of the frontier.

Normalized Deferral Curve. Figure 7 reports router accuracy and cost normalized to each router’s best-performing model, point (100%, 100%) on the plot. The upper-left quadrant represents the ideal case—higher accuracy with lower cost by leveraging smaller models. In practice, most routers cluster near the baseline (100% cost, 100% accuracy), suggesting they over-rely on the strongest model and miss opportunities to defer to cheaper alternatives. Notably, NIRT-BERT illustrates inefficiency, reaching only baseline-level accuracy while incurring 378% of the cost.

By contrast, routers such as vLLM-SR and CARROT achieve meaningful savings: roughly 35% lower cost with under 2% accuracy degradation. These cases show routing can indeed improve efficiency when smaller models are effectively utilized. Overall, the figure highlights a clear trade-off—higher accuracy often comes with higher cost—while also pointing to promising directions for designing routers that move closer to the ideal frontier.

Optimality Score. Figure 8 highlights the inherent trade-off between routing accuracy and cost. In practice, routers that achieve higher accuracy typically do so at the expense of a higher cost ratio, since they defer more often to large, expensive models. This behavior lowers their optimal selection ratio, i.e., the frequency with which they choose the most efficient model for each query. This pattern is most apparent in the binary routers such as RouteLLM. By design, these routers face a sharp trade-off: they achieve higher accuracy by routing more queries to the stronger model, which drives up cost. In contrast, multi-model routers have a more flexible pool, and while the general trend still holds, we see greater variability depending on pool composition and routing strategy. Among non-binary routers, RouterDC stands out with the lowest cost ratio and highest optimal selection ratio, but this comes at the cost of poor overall accuracy. At the other extreme, MIRT-BERT achieves strong accuracy (close to 77% of its optimal accuracy) but requires nearly five times the optimal cost, placing it closer to the “high-cost high-accuracy” region of the trade-off frontier. In other words, while some routers are closer to the efficiency frontier than others, none simultaneously combine low cost and high accuracy. Overall, our findings indicate that current routing methods have learned to leverage large models to boost performance, but remain inefficient at recognizing when smaller models are sufficient. This creates a clear opportunity for future work: developing routers that can balance accuracy and efficiency by selectively deferring to large models only when necessary.

Robustness and Latency. Given that user prompts are often noisy, we further assess router sensitivity and robustness. The detailed evaluation is illustrated in Appendix G. We define robustness as $1 -$ the proportion of changed selections. As shown in Figure 9, robustness is uniformly low across both commercial routers and academic routers such as vLLM-SR, MIRT-BERT, and NIRT-BERT. A key reason is that these academic routers depend on BERT to compute latent representations, and BERT is known to be sensitive to surface-level perturbations (Jin et al., 2020). These findings highlight the importance of applying prompt engineering techniques to mitigate noisy queries.

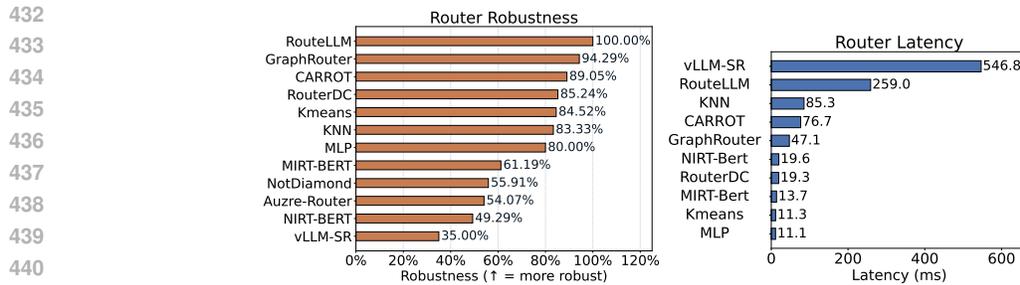


Figure 9: Router robustness and latency comparison.

Furthermore, Figure 9 reports the end-to-end latency of routers on a single A100 GPU, measured from the time a request is received to the output of the model selection result. Among all methods, vLLM-SR and RouteLLM exhibit significantly higher latency because they rely on the OpenAI embedding API, which introduces additional network delays. In contrast, other routers consistently maintain sub-100ms latencies. Since the LLM router lies on the critical path of the end-to-end systems, our results provide new insights for industrial deployment: while LLM routers can optimize accuracy and cost, they also introduce non-negligible overhead that may even compromise service-level objectives (SLOs).

Generation Length We also report the average generation length and price per M output tokens for each router in Table 8. GPT-5 and Azure-Router rely more heavily on reasoning models, leading to substantially longer generations than most other routers. Longer answers are not inherently better; this metric simply offers an additional axis for users to trade off concise versus detailed responses.

6.3 RESULTS BY DIFFICULTY LEVELS

Table 6 shows the accuracy–cost tradeoff across easy, medium, and hard queries. Most routers exceed 89% accuracy on easy questions, but accuracy drops sharply on medium and especially hard ones (often below 10%), confirming that our difficulty bands are non-trivial. GPT-5 achieves the highest accuracy at every level but at a much higher cost. At the same time, routers such as Azure-Router and vLLM-SR offer competitive accuracy on easy/medium queries at lower cost. This contrast reveals substantial headroom on hard questions and clear efficiency–accuracy tradeoffs across methods. We also observe that some routers (e.g., GPT-5, Azure-Router, MIRT-BERT, NotDiamond) allocate significantly more budget to Hard than Easy queries, while others show almost flat cost across difficulty, suggesting room for more difficulty-aware routing strategies.

6.4 ADDITIONAL RESULTS ON LONGBENCH-V2

We provide additional results on LongBench-v2 in Appendix E. On this long-context suite, GPT-5 attains the highest accuracy, while Azure-Router and NIRT-BERT offer competitive performance at substantially lower cost, illustrating clear accuracy–cost trade-offs in the long-context regime.

6.5 INSIGHTS FROM THE FINAL ROUTERARENA LEADERBOARD.

Our evaluation produces the router leaderboard shown in Table 2. The leaderboard consists of six ranking scores, and the overall ranking is determined by averaging across them. We highlight two key findings: (1) Commercial routers do not necessarily outperform open-source routers. For example, GPT-5 ranks #7 due to its restricted model pool, and NotDiamond ranks #12 because it frequently selects expensive models. (2) No router ranks at the top across all metrics, reflecting the inherent trade-offs in router design.

For developers and researchers, the findings highlight key deficiencies in current routing methods and point toward clear directions for designing the next generation of routers. The results show that all existing routers fall short of the oracle’s achievable performance, primarily because they are inefficient at recognizing when smaller, cheaper models are sufficient for a given query. Future work should focus on closing this performance gap. Moreover, the high latency and poor robustness of certain routers open new avenues of research beyond the traditional cost-versus-accuracy trade-off. Developers can use the platform’s automated framework to submit and benchmark new routers against established leaders, fostering innovation and transparently tracking progress in the field.

Table 2: Ranking of routers across multiple metrics. Lower values indicate better performance.

#	Router	Arena Rank	Optimal-selection-ratio Rank	Optimal-cost-ratio Rank	Optimal-acc-ratio Rank	Robustness Rank	Latency Rank	Average
1	Azure-Router	2	3	3	5	9	-	4.40
2	RouteLLM	9	1	1	10	1	8	5.0
3	MLP	10	4	5	8	6	1	5.67
4	MIRT-BERT	1	9	7	1	7	2	4.50
5	vLLM-SR	4	8	8	3	11	9	7.17
6	RouterDC	12	2	2	11	4	3	5.67
7	GPT-5	5	-	-	-	-	-	5
8	GraphRouter	7	6	4	9	2	5	5.50
9	CARROT	3	10	9	4	3	6	5.83
10	NIRT-BERT	6	7	10	2	10	4	6.50
11	KNN	11	5	6	7	5	7	6.83
12	NotDiamond	8	11	11	6	8	-	8.80

7 RELATED WORK

LLM Router. With the increasing availability of specialized models that can surpass even the most capable general-purpose LLMs in specific domains, both academia and industry have been actively exploring how to build LLM routers. In industry, several systems have emerged. Martian Router (WithMartian, 2025) proposed the idea of model mapping, while Storytell (Storytell.ai, 2025) categorizes user queries and routes them to the best-performing models. Other companies also seek to find the optimal model for user’s tasks by balancing performance and cost (NotDiamond, 2025; RequestyAI, 2025; OpenAI, 2025). Recent academic efforts have also begun to emerge. GraphRouter (Feng et al., 2025a) leverages graph neural networks, and Router-R1 (Zhang et al., 2025a) employs reinforcement learning. The growth of open-source solutions underscores the need for effective router evaluation (Somersstep et al., 2025; Song et al., 2025; Feng et al., 2025b).

LLM Router Benchmarks. RouterBench (Hu et al., 2024) introduces a large-scale dataset consisting of over 405k inference outcomes from representative LLMs. RouterEval (Huang et al., 2025b) collects performance results from 8,500 LLMs across 12 widely used benchmarks. Fusion-Bench (Feng et al., 2025b) covers 14 tasks across five domains and leverages 20 open-source LLMs. Other benchmarks have also contributed to this line of work by using different data collection methods (Kassem et al., 2025; Mei et al., 2025). All of these benchmarks adopt an *offline* design: they release a large frozen corpus of model outputs that is ideal for cheap, fully reproducible comparison of routing algorithms under a fixed model pool. By contrast, RouterArena is explicitly designed as a *live leaderboard* that includes both academic and commercial routers. Our live leaderboard by nature encourages router designers to explore evolving and specialized models, making routing decisions more meaningful in practice. Meanwhile, RouterArena improves on prior router benchmarks by covering more tasks and domains, and by providing a dataset with empirically calibrated difficulty together with more comprehensive metrics.

8 CONCLUSION

We introduce ROUTERARENA, the first open platform for comprehensive router comparison. Our platform features a principled dataset with broad domain coverage and varying difficulty levels, an extensive set of evaluation metrics, and an automated framework to maintain a live leaderboard. Initial evaluations of 12 routers reveal a significant trade-off between accuracy and cost, showing that no single router is universally optimal. Commercial routers tend to achieve higher accuracy at a much greater expense, while open-source routers often present more cost-efficient solutions. Overall, our findings indicate that current routers are inefficient at leveraging cheaper models when appropriate, highlighting a clear opportunity for future work.

ETHICS STATEMENT

All authors have read and will adhere to the ICLR Code of Ethics (<https://iclr.cc/public/CodeOfEthics>). Our evaluation dataset was constructed by aggregating and sampling from publicly available and open-source datasets. To ensure broad topic coverage while avoiding potentially sensitive subjects, we based our domain selection on the Dewey Decimal Classification system, explicitly excluding the category of religion. The difficulty level of each query was annotated using an "LLM-as-Judge" approach. We acknowledge that this process, along with the use of existing datasets, may introduce or perpetuate biases inherent in the source data and the annotator model. We have made the dataset and our collection methodology public to allow for further inspection and bias analysis by the community.

REPRODUCIBILITY STATEMENT

We are committed to ensuring the full reproducibility of our work. All components of the ROUTER-ARENA platform, including the dataset, evaluation framework, and code, will be made publicly available.

(1) Dataset: The principles and detailed process for our dataset construction are described in Section 3. This includes domain coverage strategy, difficulty level annotation, and our deduplication process. Further details on the dataset schema, composition, and examples are provided in Appendix C. (2) Evaluation: Our five evaluation metrics are precisely defined in Section 4, and the formula for the composite leaderboard score is detailed in Section 5.1. The automated evaluation framework is described in Section 5.2. (3) Experiments: The specific academic and commercial routers evaluated are listed in Section 6.1.1. The exact model pools used for each router are provided in Appendix B. For all academic routers, we followed the training procedures and configurations specified in their original open-source implementations.

The public release of our complete framework will enable researchers to replicate our results and evaluate new routers on the leaderboard.

REFERENCES

- Benjamin S Bloom et al. Taxonomy of. *Educational Objectives*, 250, 1956.
- Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.
- Shuhao Chen, Weisen Jiang, Baijiong Lin, James Kwok, and Yu Zhang. Routerdc: Query-based router by dual contrastive learning for assembling large language models. *Advances in Neural Information Processing Systems*, 37:66305–66328, 2024a.
- Shuhao Chen, Weisen Jiang, Baijiong Lin, James T. Kwok, and Yu Zhang. Routerdc: Query-based router by dual contrastive learning for assembling large language models, 2024b. URL <https://arxiv.org/abs/2409.19886>.
- Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. In *Forty-first International Conference on Machine Learning*, 2024.
- Yu-Neng Chuang, Prathusha Kameswara Sarma, Parikshit Gopalan, John Boccio, Sara Bolouki, Xia Hu, and Helen Zhou. Learning to route llms with confidence tokens. *arXiv preprint arXiv:2410.13284*, 2024.
- DeepSeek-AI. Deepseek-v3 technical report, 2025. URL <https://arxiv.org/abs/2412.19437>.
- Melvil Dewey. *A classification and subject index, for cataloguing and arranging the books and pamphlets of a library*. Brick row book shop, Incorporated, 1876.

- 594 Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian Del Car-
595 men Hipolito Garcia, Menglin Xia, Laks VS Lakshmanan, Qingyun Wu, and Victor Rühle. Best-
596 route: Adaptive llm routing with test-time optimal compute. *arXiv preprint arXiv:2506.22716*,
597 2025.
- 598 Tao Feng, Yanzen Shen, and Jiaxuan You. Graphrouter: A graph-based router for llm selections,
599 2025a. URL <https://arxiv.org/abs/2410.03834>.
- 601 Tao Feng, Haozhen Zhang, Zijie Lei, Pengrui Han, Mostofa Patwary, Mohammad Shoeybi, Bryan
602 Catanzaro, and Jiaxuan You. Fusing llm capabilities with routing data, 2025b. URL <https://arxiv.org/abs/2507.10540>.
- 604 Wirth F Ferger. The nature and use of the harmonic mean. *Journal of the American Statistical*
605 *Association*, 26(173):36–40, 1931.
- 607 Surya Narayanan Hari and Matt Thomson. Tryage: Real-time, intelligent routing of user prompts to
608 large language models. *arXiv preprint arXiv:2308.11601*, 2023.
- 610 Annette Herrmann-Werner, Tanja Festl-Wietek, Friederike Holderried, Lena Herschbach, Jan
611 Griewatz, Ken Masters, Stephan Zipfel, and Matthias Mahling. Assessing ChatGPT’s mas-
612 tery of Bloom’s Taxonomy using psychosomatic medicine exam questions: Mixed-methods
613 study. *Journal of Medical Internet Research*, 26:e52113, 2024. doi: 10.2196/52113. URL
614 <https://doi.org/10.2196/52113>.
- 615 Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt
616 Keutzer, and Shriyash Kaustubh Upadhyay. Routerbench: A benchmark for multi-llm routing
617 system, 2024. URL <https://arxiv.org/abs/2403.12031>.
- 618 Zhongzhan Huang, Guoming Ling, Yupei Lin, Yandong Chen, Shanshan Zhong, Hefeng Wu, and
619 Liang Lin. Routereval: A comprehensive benchmark for routing llms to explore model-level
620 scaling up in llms. *arXiv preprint arXiv:2503.10657*, 2025a.
- 622 Zhongzhan Huang, Guoming Ling, Yupei Lin, Yandong Chen, Shanshan Zhong, Hefeng Wu, and
623 Liang Lin. Routereval: A comprehensive benchmark for routing llms to explore model-level
624 scaling up in llms, 2025b. URL <https://arxiv.org/abs/2503.10657>.
- 625 Hugging Face. Modernbert: A new era of pretraining. [https://huggingface.co/blog/
626 modernbert](https://huggingface.co/blog/modernbert), 2025. Hugging Face Blog, accessed: Sep. 2025.
- 627 Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is bert really robust? a strong baseline for
628 natural language attack on text classification and entailment. *Proceedings of the AAAI Conference*
629 *on Artificial Intelligence*, 34(05):8018–8025, Apr. 2020. doi: 10.1609/aaai.v34i05.6311. URL
630 <https://ojs.aaai.org/index.php/AAAI/article/view/6311>.
- 631 Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Jeevesh Juneja, Congchao Wang,
632 Zifeng Wang, Alec Go, Chen-Yu Lee, Pradeep Shenoy, Rina Panigrahy, Aditya Krishna Menon,
633 and Sanjiv Kumar. Universal model routing for efficient llm inference, 2025. URL <https://arxiv.org/abs/2502.08773>.
- 634 Aly M. Kassem, Bernhard Schölkopf, and Zhijing Jin. How robust are router-llms? analysis of the
635 fragility of llm routing capabilities, 2025. URL <https://arxiv.org/abs/2504.07113>.
- 636 Yueyue Liu, Hongyu Zhang, Yuantian Miao, Van-Hoang Le, and Zhiqiang Li. Optllm: Optimal
637 assignment of queries to large language models. In *2024 IEEE International Conference on Web*
638 *Services (ICWS)*, pp. 788–798. IEEE, 2024.
- 639 Kai Mei, Wujiang Xu, Shuhang Lin, and Yongfeng Zhang. Omnirouter: Budget and performance
640 controllable multi-llm routing, 2025. URL <https://arxiv.org/abs/2502.20576>.
- 641 Microsoft. Model router for azure ai foundry (preview) concepts. [https://
642 learn.microsoft.com/en-us/azure/ai-foundry/openai/concepts/
643 model-router](https://learn.microsoft.com/en-us/azure/ai-foundry/openai/concepts/model-router), 2025. Accessed: Sep 2025.

- 648 NotDiamond. Notdiamond: Routing between ai models. [https://www.notdiamond.ai/](https://www.notdiamond.ai/about)
649 about, 2025. Accessed: Sep. 2025.
- 650
- 651 Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez,
652 M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms with preference data, 2025.
653 URL <https://arxiv.org/abs/2406.18665>.
- 654 OpenAI. Gpt-5 system card. <https://cdn.openai.com/gpt-5-system-card.pdf>,
655 August 2025. Accessed: Sep 2025.
- 656
- 657 Ulrike Padó. Question difficulty – how to estimate without norming, how to use for automated
658 grading. In Joel Tetreault, Jill Burstein, Claudia Leacock, and Helen Yannakoudakis (eds.), *Pro-*
659 *ceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*,
660 pp. 1–10, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.
661 doi: 10.18653/v1/W17-5001. URL <https://aclanthology.org/W17-5001/>.
- 662 RequestyAI. Requestyai: Unified llm gateway routing. [https://www.requesty.ai/](https://www.requesty.ai/solution/llm-routing)
663 solution/llm-routing, 2025. Accessed: Sep. 2025.
- 664
- 665 Mohinder Partap Satija. *The theory and practice of the Dewey decimal classification system*. Else-
666 vier, 2013.
- 667
- 668 Seamus Somerstep, Felipe Maia Polo, Allysson Flavio Melo de Oliveira, Prattyush Mangal, Mírian
669 Silva, Onkar Bhardwaj, Mikhail Yurochkin, and Subha Maity. Carrot: A cost aware rate optimal
670 router, 2025. URL <https://arxiv.org/abs/2502.03261>.
- 671
- 672 Wei Song, Zhenya Huang, Cheng Cheng, Weibo Gao, Bihan Xu, GuanHao Zhao, Fei Wang, and
673 Runze Wu. Irt-router: Effective and interpretable multi-llm routing via item response theory,
674 2025. URL <https://arxiv.org/abs/2506.01048>.
- 675
- 676 Storytell.ai. Storytell.ai: Dynamic llm routing. [https://web.storytell.ai/](https://web.storytell.ai/llm-router)
677 llm-router, 2025. Accessed: Sep. 2025.
- 678
- 679 Sabine Ullrich and Michaela Geierhos. Using bloom’s taxonomy to classify question com-
680 plexity. In Mourad Abbas and Abed Alhakim Freihat (eds.), *Proceedings of the 4th Inter-*
681 *national Conference on Natural Language and Speech Processing (ICNLSP 2021)*, pp. 285–
682 289, Trento, Italy, 12–13 November 2021. Association for Computational Linguistics. URL
683 <https://aclanthology.org/2021.icnls-1.34/>.
- 684
- 685 vLLM. vllm semantic router. <https://vllm-semantic-router.com/>, 2025. Accessed:
686 Sep. 2025.
- 687
- 688 Xinyuan Wang, Yanchi Liu, Wei Cheng, Xujiang Zhao, Zhengzhang Chen, Wenchao Yu, Yanjie Fu,
689 and Haifeng Chen. Mixllm: Dynamic routing in mixed large language models. *arXiv preprint*
690 *arXiv:2502.18482*, 2025.
- 691
- 692 WithMartian. Martian model router. [https://docs.withmartian.com/](https://docs.withmartian.com/martian-model-router)
693 martian-model-router, 2025. Accessed: Sep. 2025.
- 694
- 695 Haozhen Zhang, Tao Feng, and Jiaxuan You. Router-r1: Teaching llms multi-round routing and
696 aggregation via reinforcement learning, 2025a. URL [https://arxiv.org/abs/2506.](https://arxiv.org/abs/2506.09033)
697 09033.
- 698
- 699 Yi-Kai Zhang, Ting-Ji Huang, Yao-Xiang Ding, De-Chuan Zhan, and Han-Jia Ye. Model spider:
700 Learning to rank pre-trained models efficiently. *Advances in Neural Information Processing Sys-*
701 *tems*, 36:13692–13719, 2023.
- 702
- 703 Yi-Kai Zhang, De-Chuan Zhan, and Han-Jia Ye. Capability instruction tuning: A new paradigm for
704 dynamic llm routing. *arXiv preprint arXiv:2502.17282*, 2025b.
- 705
- 706 Xinyu Zhao, Guoheng Sun, Ruisi Cai, Yukun Zhou, Pingzhi Li, Peihao Wang, Bowen Tan, Yexiao
707 He, Li Chen, Yi Liang, et al. Model-glue: Democratized llm scaling for a large model zoo in the
708 wild. *Advances in Neural Information Processing Systems*, 37:13349–13371, 2024.

702 Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchandran.
703 Embedlm: Learning compact representations of large language models, 2024a. URL <https://arxiv.org/abs/2410.02223>.
704
705 Richard Zhuang, Tianhao Wu, Zhaojin Wen, Andrew Li, Jiantao Jiao, and Kannan Ramchan-
706 dran. Embedlm: Learning compact representations of large language models. *arXiv preprint*
707 *arXiv:2410.02223*, 2024b.
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755

A THE USE OF LLMs

In this work, Large Language Models (LLMs) were used in two distinct capacities: as a core component of our research methodology and as a general-purpose writing aid.

LLM-as-Judge for Data Annotation: A significant use of an LLM was in the construction of our evaluation dataset. As detailed in Section 3, we employed DeepSeek-V3.1 as an “LLM-as-Judge” to automatically annotate the cognitive level of each query according to Bloom’s taxonomy. This automated approach allowed us to systematically label a large and diverse set of questions. The specific prompt used for this annotation process is provided in Appendix D.

Writing Assistance: An LLM was also utilized as a general-purpose tool to assist in the writing process. Its role was limited to proofreading the manuscript for grammatical errors, improving clarity, and ensuring stylistic consistency. The LLM was not used for research ideation, conducting experiments, or writing the core scientific contributions of the paper.

B MODEL POOLS BY ROUTER

Table 3: Model pools used by different routers.

Router	Model Pool
RouterBench	WizardLM/WizardLM-13B-V1.2; claude-instant-v1; claude-v1; claude-v2; gpt-3.5-turbo-1106; gpt-4-1106-preview; meta/codellama-34b-instruct; meta/llama-2-70b-chat; mistralai/mistral-7b-chat; mistralai/mixtral-8x7b-chat; zero-one-ai/Yi-34B-Chat
GraphRouter	meta-llama/llama-3-8b-instruct; mistralai/mixtral-8x7b-chat; nousresearch/nous-34b-chat; meta/llama-2-7b-chat; mistralai/mistral-7b-chat; meta/llama-3-70b-chat; meta/llama-3-turbo-8b-chat; meta/llama-3-turbo-70b-chat; meta/llama-3.1-turbo-70b-chat; qwen/qwen-1.5-72b-chat
Universal	WizardLM/WizardLM-13B-V1.2; claude-instant-v1; claude-v1; claude-v2; gpt-3.5-turbo-1106; gpt-4-1106-preview; meta/codellama-34b-instruct; meta/llama-2-70b-chat; mistralai/mistral-7b-chat; mistralai/mixtral-8x7b-chat; zero-one-ai/Yi-34B-Chat
CarrotRouter	aws-claude-3-5-sonnet-v1; aws-titan-text-premier-v1; openai-gpt-4o; openai-gpt-4o-mini; wxai-granite-3-2b-instruct-8k-max-tokens; wxai-granite-3-8b-instruct-8k-max-tokens; wxai-llama-3-1-70b-instruct; wxai-llama-3-1-8b-instruct; wxai-llama-3-2-1b-instruct; wxai-llama-3-2-3b-instruct; wxai-llama-3-3-70b-instruct; wxai-mixtral-8x7b-instruct-v01; wxai-llama-3-405b-instruct
RouterDC	mistralai/Mistral-7B-v0.1; meta-math/MetaMath-Mistral-7B; itpossible/Chinese-Mistral-7B-v0.1; HuggingFaceH4/zephyr-7b-beta; cognitivecomputations/dolphin-2.6-mistral-7b; meta-llama/llama-3-8b-instruct; cognitivecomputations/dolphin-2.9-llama3-8b
IRT-Router	glm_4_air; glm_4_flash; glm_4_plus; gpt_4o; gpt_4o_mini; gpt_4o_mini_cot; deepseek_coder; deepseek_chat; qwen25_32b_int4; qwen25_7b_instruct; qwen25_72b_instruct; qwq_32b_preview; qwen25_math_7b_instruct; llama31_8b_instruct; llama31_70b_instruct; llama31_405b_instruct; mixtral_8x7b_instruct; mistral_7b_instruct_v02; ministral_8b_instruct_2410; gemini15_flash; claude35_haiku20241022
RouteLLM	openai-gpt-4o; mixtral_8x7b_instruct

We provide the model pool used by each router here.

C DATASETS DETAILS

C.1 USE OF LLM-AS-A-JUDGE

We sampled 450 queries from the dataset by stratifying along both the domain and Bloom level, ensuring that each combination was proportionally represented. Eighteen volunteers (undergraduate and graduate students) were asked to assign a Bloom level to each query under the same instructions as used for the DEEPSEEK-V3.1 judge. For each query, we compare the human label with the LLM’s prediction. We observe 54.9% exact agreement and 76.7% agreement within ± 1 Bloom

level, indicating that most disagreements occur between adjacent levels in this inherently subjective taxonomy. Since Bloom labels are used only to ensure cognitive-skill coverage during dataset construction—while all evaluation difficulty is defined empirically via accuracy over 42 LLMs—this level of agreement is sufficient for our purposes.

C.2 DATASET COMPOSITION AND SCHEMA

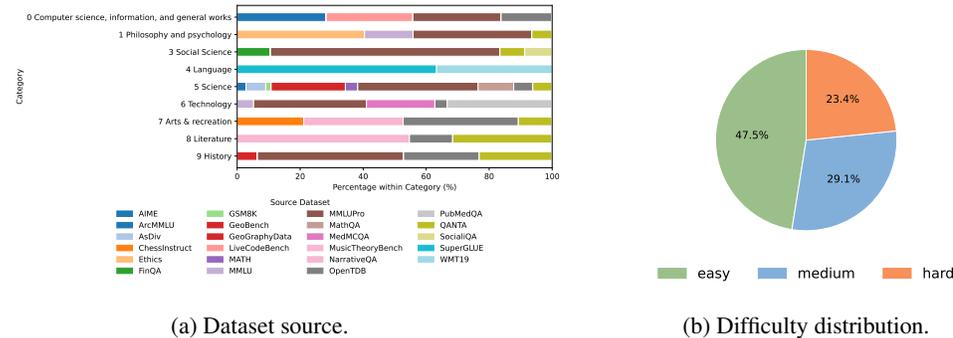


Figure 10: Overview of dataset composition: (a) data sources and (b) difficulty distribution.

Figure 10 illustrates the domain coverage of our dataset across the nine Dewey Decimal categories. Each horizontal bar represents the relative contribution of different source datasets within a category. This distribution highlights the systematic integration of multiple datasets to achieve a more balanced representation of both general-purpose and highly specialized domains.

Table 4: Overview of dataset columns

Column	Description	Example
Category	Bloom’s taxonomy high-level class	9 History
Sub Category	Bloom’s taxonomy sub-class	02 Library and information sciences
Dataset Name	Source dataset	ArcMMLU
Global Index	Unique instance ID	ArcMMLU_114
Context	Supporting passage (if any)	Sasha decided to watch TV and get some food
Question	Input question	What is the capital of France?
Options	Multiple-choice options	[“10”, “20”, “30”, “40”]
Answer	Ground-truth answer	Paris
Bloom Level	Bloom’s taxonomy difficulty level	Understanding

Table 4 shows the detailed dataset columns.

C.3 DATASET EXAMPLES

Figure 11 shows some dataset examples.

C.4 MODEL LIST FOR EMPIRICAL DIFFICULTY LABELING

We use the following 42 models to compute the empirical difficulty labels shown in Figure 5. These models are drawn from the model pools of the routers we benchmark, and were chosen to cover a broad range of parameter scales, architectures, and providers. This diversity ensures that queries elicit a wide spread of accuracies across models, which we indeed observe in their scores and use as the basis for our empirical difficulty measure.

Table 5: These are the 42 models used for empirical difficulty labeling. Models span across a range of sizes and performances, showcasing that ROUTERARENA could distinguish LLMs by providing diverse questions of difficulty.

Rank	Model	Score [%]
1	gpt-5	74.04
2	claude-3-5-sonnet	67.73
3	deepseek/deepseek-chat	66.86
4	gpt-4o	66.35
5	llama-3-1-405b-instruct	65.43
6	glm-4-plus	62.02
7	gpt-4-1106-preview	61.91
8	meta-llama_Meta-Llama-3.1-70B-Instruct-Turbo	60.64
9	Qwen/Qwen2.5-32B-Instruct-GPTQ-Int4	60.51
10	Qwen/Qwen2.5-72B-Instruct	59.98
11	meta-llama_Llama-3.1-70B-Instruct	59.89
12	Qwen/QwQ-32B	59.81
13	gpt-4o-mini	59.22
14	meta-llama_Meta-Llama-3-70B-Instruct	58.16
15	google/gemini-flash-1.5	57.21
16	glm-4-air	54.63
17	Qwen/Qwen1.5-72B-Chat	49.86
18	Qwen/Qwen2.5-7B-Instruct	47.70
19	glm-4-flash	47.62
20	mistralai/Mistral-8x7B-Instruct-v0.1	47.01
21	01-ai/Yi-34B-Chat	45.54
22	llama-3-1-8b-instruct	45.23
23	NousResearch/Nous-Hermes-2-Yi-34B	45.01
24	meta-llama_Llama-3.1-8B-Instruct	44.24
25	meta-llama_Llama-2-70b-chat-hf	39.82
26	meta/llama-2-70b-chat	39.49
27	mistralai/Mistral-8B-Instruct-2410	32.87
28	llama-3-2-3b-instruct	32.55
29	cognitivecomputations/dolphin-2.9-llama3-8b	32.35
30	Qwen/Qwen2.5-32B	32.26
31	mistralai/Mistral-7B-Instruct-v0.2	31.83
32	meta-llama.Llama-3-8b-instruct	30.45
33	cognitivecomputations/dolphin-2.6-mistral-7b	29.98
34	mistralai/Mistral-7B-Instruct-v0.3	28.05
35	meta-llama_Meta-Llama-3-8B-Instruct	23.69
36	WizardLM/WizardLM-13B-V1.2	23.03
37	llama-3-2-1b-instruct	22.10
38	meta-llama_Llama-2-7b-chat-hf	20.50
39	meta/codellama-34b-instruct	20.42
40	codellama/CodeLlama-34b-Instruct-hf	19.31
41	meta-llama_Meta-Llama-2-7B	18.92
42	meta-llama_Meta-Llama-2-7B-hf	18.39

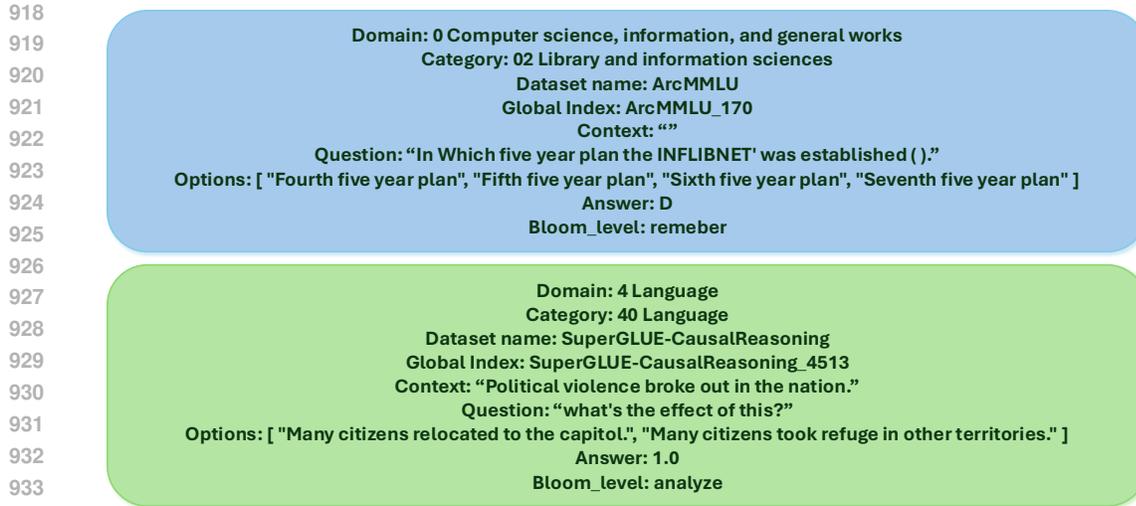


Figure 11: Dataset examples.

D LLM PROMPTS

D.1 BLOOM TAXONOMY PROMPT

We used the following prompt for classifying each question into different Bloom Levels:

Instruction: You are an evaluator tasked with classifying questions by cognitive difficulty using **Bloom’s Taxonomy (Revised 2001)**. Bloom’s Taxonomy defines six levels of cognitive processes:

1. **Remember** – Recall or recognize facts, terms, or concepts. 2. **Understand** – Explain, summarize, interpret, or demonstrate comprehension. 3. **Apply** – Use learned knowledge to solve problems in new or routine situations. 4. **Analyze** – Break down information, examine parts, relationships, or underlying causes. 5. **Evaluate** – Make judgments or decisions based on evidence, criteria, or standards. 6. **Create** – Put elements together to form a new structure, idea, or product.

Your Task: Given the details of a question, determine *which Bloom’s level best represents the cognitive process required to answer it*.

- Output the result in a **structured JSON block** with no additional text. - If multiple levels might apply, choose the *highest* level required.

Output Format:

```
{
  "bloom_level": "<Remember | Understand | Apply | Analyze |
  Evaluate | Create>"
}
```

Input Fields:

- Dataset name: {dataset_name}
- Dataset classification (DDC category): {dataset_category}
- Question: {question}
- Question Index: {question_index}
- Context (if any): {context}
- Options (if MCQ): {options}
- Answer: {answer}

Table 6: Router performance by difficulty level (accuracy and cost per 1k queries).

Router	Overall	Easy	Medium	Hard
GPT-5	74.0% (\$14.02)	95.1% (\$5.68)	68.6% (\$14.80)	27.5% (\$35.73)
Azure-Router	68.1% (\$0.54)	93.3% (\$0.30)	59.5% (\$0.63)	17.9% (\$1.05)
NotDiamond	68.0% (\$9.34)	92.5% (\$6.34)	61.9% (\$10.29)	15.1% (\$15.16)
vLLM-SR	67.3% (\$1.67)	95.3% (\$1.56)	57.9% (\$1.70)	8.7% (\$1.87)
CARROT	67.2% (\$2.06)	95.0% (\$1.79)	58.0% (\$2.26)	9.0% (\$2.36)
MIRT-BERT	66.9% (\$0.15)	95.9% (\$0.10)	56.8% (\$0.17)	7.1% (\$0.26)
NIRT-BERT	62.0% (\$0.69)	93.7% (\$0.53)	47.0% (\$0.78)	4.7% (\$0.90)
MLP	61.6% (\$4.83)	93.2% (\$3.54)	46.6% (\$5.74)	4.6% (\$6.52)
KNN	58.7% (\$4.27)	89.2% (\$3.08)	43.6% (\$5.13)	4.5% (\$5.77)
Graphrouter	57.0% (\$0.34)	89.7% (\$0.25)	39.3% (\$0.39)	2.4% (\$0.46)
RouteLLM	47.0% (\$0.27)	79.8% (\$0.20)	24.4% (\$0.33)	2.0% (\$0.32)
RouterDC	32.0% (\$0.07)	54.9% (\$0.06)	15.3% (\$0.08)	2.2% (\$0.09)

D.2 EVALUATION PROMPT

The following is an example of an evaluation prompt we use:

Please read the following multiple-choice questions and consider which option best follows deontological ethics.\n\n Context: {Context}\n\n Question: {Question}\n\n Options: \n{Options}\n\n Provide the correct letter choice in \boxed{X} , where X is the correct letter choice.
Keep the explanation or feedback within 3 sentences.

E ADDITIONAL RESULTS AND TABLES:

Long-context routing. Table 7 reports router performance on the LongBench-v2 dataset. We randomly sampled 100 queries of short and medium length from the dataset. GPT-5 attains the highest accuracy (71%) but also the highest cost per 1k queries. Azure-Router, MIRT-BERT, and NIRT-BERT achieve reasonably strong accuracy (60–67%) at one order of magnitude lower cost, whereas methods like MLP, KNN, NotDiamond, and vLLM-SR are substantially more expensive for comparable or worse accuracy. RouteLLM cannot be evaluated in this setting because its text encoder only supports inputs up to 8,192 tokens, which excludes many of our long-context queries.

Generation length and cost. Table 8 summarizes average generation length and cost per million output tokens. GPT-5 produces the longest responses and is also the most expensive per token, while Azure-Router, NIRT-BERT, and MIRT-BERT generate shorter outputs at much lower cost. In contrast, heuristic baselines such as MLP, KNN, and CARROT have relatively short generations yet very high token-level costs, indicating that they are inefficient choices when generation cost is a primary concern.

F LEADERBOARD

Figure 12 presents the spider plot of RouterArena, which compares six routing methods (CARROT, RouterDC, GraphRouter, MIRT-BERT, NIRT-BERT, and RouteLLM) across five evaluation dimensions: Arena Score, Cost-ratio Score, Optimal-acc Score, Latency Score, and Robustness Score. Each axis indicates higher performance in the outward direction, allowing a direct visualization of trade-offs. For example, CARROT achieves strong performance in Arena and Latency Scores, while RouterDC excels in Cost-ratio Score.

Table 7: RouterArena routers—long-context performance and cost. RouteLLM cannot be evaluated on this subset because its text encoder only supports inputs up to 8,192 tokens.

Router	Acc (%)	Cost / 1K Queries (USD)
GPT-5	71	45.70
NotDiamond	70	59.30
Azure-Router	67	9.54
NIRT-BERT	60	6.60
RouteLLM	–	–
Graphrouter	42	23.60
vLLM-SR	42	64.80
CARROT	25	4.10
RouterDC	12	3.60
MIRT-BERT	60	7.50
MLP	65	286.70
KNN	69	300.30

Table 8: RouterArena routers—average generation length and cost.

Router	Avg. Gen. Length	Cost / M Tokens (USD)
GPT-5	970.65	10.00
NotDiamond	390.05	9.18
Azure-Router	270.90	1.79
NIRT-BERT	215.82	1.33
RouteLLM	140.24	0.68
Graphrouter	132.24	0.81
vLLM-SR	109.27	9.03
CARROT	105.56	12.58
RouterDC	102.83	0.20
MIRT-BERT	89.16	1.00
MLP	81.06	29.75
KNN	79.80	26.31

G ROBUSTNESS EVALUATION

Specifically, we assign equal weight to the four noisy variants described in the definition. For each category and its subcategories, we randomly sampled 5% of the entire dataset, which is 420 examples and generated perturbed queries using GPT-4o. The exact prompt we used to generate perturbed queries is as follows:

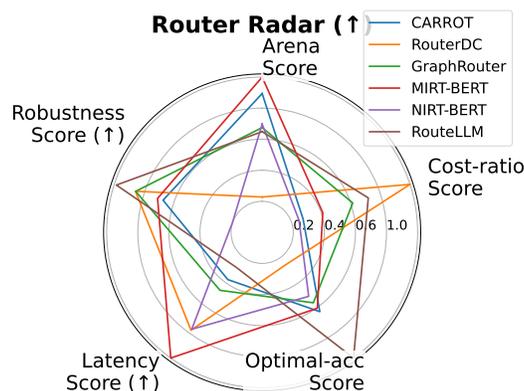


Figure 12: The spider plot of ROUTERARENA

1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133

You are an assistant that performs extremely strong and highly disruptive text transformations for robustness stress testing. You must follow the selected mode exactly. Transformations must be intense, obvious, and aggressive. Always preserve the requested structure, but you may heavily distort surface wording. Do NOT shorten or expand the text beyond the allowed limits.

There are four transformation modes:

PARAPHRASE (MAXIMAL TRANSFORMATION MODE) Rewrite the text so heavily that almost no original sentences or phrasing remain. You may reorder ideas, split or merge sentences, and aggressively substitute expressions. The meaning must stay intact, but the wording should look almost unrecognizable from the input. Keep length approximately the same (± 15

GRAMMATICAL RECONSTRUCTION (DEEP REWRITE MODE) Rebuild the text with fully restructured, polished, and elegant grammar. You may reorder clauses, shift emphasis, modernize tone, and transform the writing style completely. All information must remain, but the writing should feel dramatically improved and profoundly different from the original. Keep the length the same (± 10)

SYNONYM SATURATION (ULTRA-DENSE SUBSTITUTION MODE) Replace nearly every content word (nouns, verbs, adjectives, adverbs) with strong, less common, or surprising synonyms. You may alter idioms, swap multiword phrases with equivalent expressions, and push semantic boundaries while preserving meaning. The output should visually look almost entirely changed. Maintain length (± 10)

INTENTIONAL CORRUPTION (HEAVY DEGRADATION MODE) Inject a very high density of diverse, realistic, and chaotic textual corruptions: misspellings, keyboard slips, letter swaps, missing characters, inserted random characters, spacing glitches, partial word breakage, and inconsistent capitalization. Meaning must remain barely decipherable but still recoverable. Maintain approximate length (± 10)

Input format:

MODE: <Paraphrase — Grammar — Synonyms — Typos>

TEXT: <user_provided_text>

Output format (strict):

=== TRANSFORMED TEXT === <extremely modified text>

=== NOTES === <1–2 very short descriptions of the corruption intensity and strategy>

Do not output anything else.

H DISCUSSION ABOUT HYPERPARAMETER(c_{min} , c_{max} , β)

In our framework, c_{max} and c_{min} are not intended to be static global constants; instead, they are tunable for flexibility. To prevent the most expensive router from being normalized to zero, for now, we set c_{max} and c_{min} to the prices of the most expensive and cheapest models on the market. However, whenever a model’s actual cost really falls below c_{min} or exceeds c_{max} , the only thing we need to do is to adjust the c_{min} and c_{max} , and re-calculate the ArenaScore so that all normalized costs remain within (0, 1). Note that this doesn’t require re-running the experiment.

The parameter β is also adjustable to modulate how much weight is assigned to cost compared to accuracy. Specifically, we made the β tunable on our leaderboard for users. Here we give three example settings:

- Example 1: $\beta = 0.01$, Accuracy : Cost = 100, accuracy-dominated ranking for users who do not consider cost to be a key constraint.
- Example 2: $\beta = 0.1$, Accuracy : Cost = 10, trade-off balanced ranking for users who primarily care about accuracy but still operate under budget constraints.

- Example 3: $\beta = 1$, Accuracy : Cost = 1, equally weighted ranking for users who have a limited budget.

Router	Rank & Score @ $\beta=0.01$	Rank & Score @ $\beta=0.1$	Rank & Score @ $\beta=1$	Accuracy	Norm.Cost
GPT-5	1 (0.7282)	5 (0.6272)	12 (0.3688)	0.7428	0.2453
Azure-Router	2 (0.6780)	2 (0.6640)	2 (0.6010)	0.6798	0.5386
MIRT-BERT	3 (0.6731)	1 (0.6729)	1 (0.6718)	0.6731	0.6705
CARROT	4 (0.6682)	3 (0.6386)	6 (0.5219)	0.6720	0.4266
vLLM-SR	5 (0.6633)	4 (0.6385)	5 (0.5368)	0.6665	0.4494
NotDiamond	6 (0.6565)	8 (0.5935)	11 (0.3998)	0.6651	0.2858
RouteLLM	7 (0.6175)	9 (0.5800)	10 (0.4440)	0.6224	0.3451
NIRT-BERT	8 (0.6151)	6 (0.6083)	4 (0.5764)	0.6159	0.5416
MLP	9 (0.6143)	10 (0.5780)	9 (0.4449)	0.6191	0.3472
GraphRouter	10 (0.6070)	7 (0.6054)	3 (0.5976)	0.6072	0.5884
KNN	11 (0.5867)	11 (0.5577)	8 (0.4464)	0.5905	0.3588
RouterDC	12 (0.3362)	12 (0.3522)	7 (0.4627)	0.3344	0.7507

Figure 13: Recommended β value for different user needs.

I PRICE OF MODELS

Table 9 shows the price we used for the self-served model. It was used by together.ai at the time the experiment was conducted, and the price reflects the hardware requirement for the increasing model size.

Table 9: Dense and MoE large language model size and price per million tokens.

Dense Models		Mixture-of-Experts Models	
Model Size	Price	Model Size	Price
Up to 4B	\$0.10	Up to 56B	\$0.60
4.1B – 8B	\$0.20	56.1B – 176B	\$1.20
8.1B – 21B	\$0.30	176.1B – 480B	\$2.40
21.1B – 41B	\$0.80		
41.1B – 80B	\$0.90		
80.1B – 110B	\$1.80		