LEARNING AGENT ROUTING FROM EARLY EXPERIENCE

Yimin Wang[*2,4] **Jiahao Qiu**[*1] **Xuan Qi**[3] **Xinzhe Juan**[2,4] **Jingzhe Shi**[3]
Zelin Zhao[6] **Hongru Wang**[5] **Shilong Liu**[†1] **Mengdi Wang**[†1]

[1]AI Lab, Princeton University [2]University of Michigan
[3]Institute for Interdisciplinary Information Sciences (IIIS), Tsinghua University
[4]Shanghai Jiao Tong University [5]University of Edinburgh [6]King's College London

ABSTRACT

LLM agents achieve strong performance on complex reasoning tasks but incur high latency and compute cost. In practice, many queries fall within the capability boundary of cutting-edge LLMs and do not require full agent execution, making effective routing between LLMs and agents a key challenge. We study the problem of routing queries between lightweight LLM inference and full agent execution under realistic cold-start settings. To address this, we propose BoundaryRouter, a training-free routing framework that uses early behavioral experience and rubric-guided reasoning to decide whether to answer a query with direct LLM inference or escalate to an agent. BoundaryRouter builds a compact experience memory by executing both systems on a shared seed set and retrieves similar cases at inference time to guide routing decisions. To evaluate this method, we introduce RouteBench, a benchmark covering in-domain, paraphrased, and out-of-domain route settings. Experiments show that BoundaryRouter reduces inference time by 60.6% compared to the agent while improving performance by 28.6% over direct LLM inference, outperforming prompt-based and retrieval-only routing by an average of 37.9% and 8.2%, respectively.

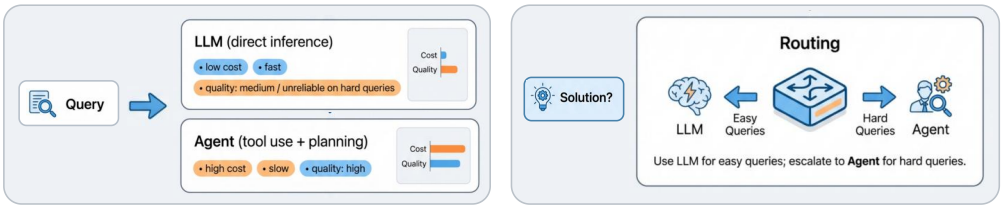


Figure 1: Motivation and overview of routing. Direct LLM inference is fast and low-cost but can be unreliable on harder queries, while full agent execution is slower and more expensive. A router dispatches each query to the appropriate system, using the LLM for easy cases and escalating to the agent when needed to achieve a better accuracy–latency trade-off.

1 INTRODUCTION

Large language model agents have recently emerged as a powerful paradigm for solving tasks that require reasoning, planning, and interaction with external environments (Zhou et al., 2025; Hu et al., 2025; Zhang et al., 2025b; Qiu et al., 2025c; H2O.ai, 2025; Team, 2025; Qiu et al., 2025b). By combining language understanding with tool use, retrieval, and long-term memory, these agents show strong adaptability across a wide range of domains, from code generation to specialized scientific and scholarly domains (Yang et al., 2024; Qiu et al., 2025a; Li et al., 2025; Wang et al., 2025; Qiu et al., 2025d; Ding et al., 2025b). However, not every task requires the complex capabilities of agents, such as multi-step reasoning or long-context management (see Fig. 1). Contemporary LLMs, trained

[*] Equal contribution.
[†] Corresponding authors.

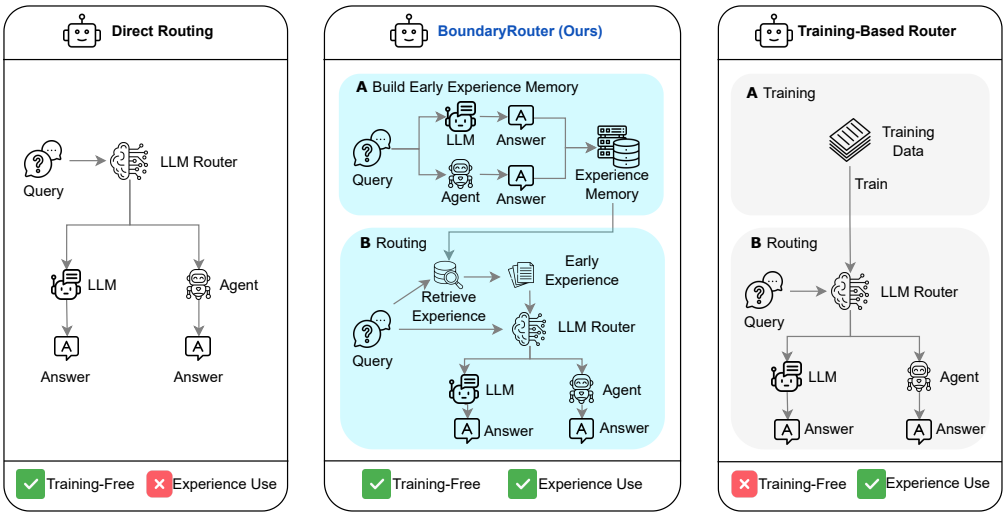


Figure 2: Comparison of routers. *Left*: Direct routing uses an LLM router to choose between direct LLM inference and full agent execution, but does not leverage experience. *Right*: Training-based routing learns a router from labeled training data, enabling experience use but requiring supervision. *Middle*: BoundaryRouter (ours) is training-free yet experience-driven: it first builds an early experience memory by running both the LLM and the agent on a small shared seed set, then retrieves similar experiences at test time to guide routing decisions.

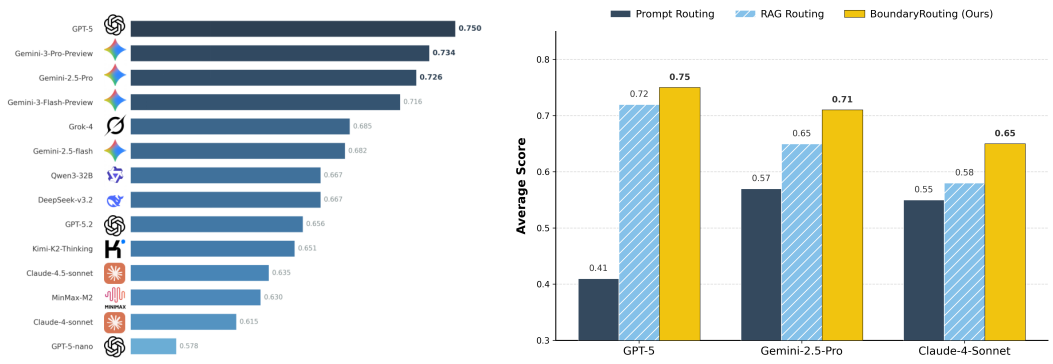
on web-scale corpora and in many cases coupled with web search tools (for example, GPT with online search in production APIs (OpenRouter, 2025)), can already solve a wide range of factual and well-structured queries with a single forward inference while with much lower computational and latency cost than a multi-step agent.

Hence, the central challenge now is to characterize the intelligence boundary of LLMs, enabling direct LLM inference within the boundary and escalating to agent only for tasks that exceed it. LLM query routing offers a practical way to probe this boundary by dynamically dispatching the query to models of varying quality and cost. Yet, existing research primarily focuses on routing exclusively among LLMs or among agents (Zhang et al., 2025a;d; Yue et al., 2025; Liu et al., 2025b), leaving the hybrid routing problem between LLMs and agents largely unexplored.

To address this issue, we propose **BoundaryRouter**, a training-free query routing method that efficiently combines direct LLM inference with agentic execution through early experience and structured reasoning. A critical constraint in real-world routing is the cold-start problem: we often lack prior performance data (ground truth) for incoming queries and therefore cannot train a supervised router. BoundaryRouter addresses this by utilizing *early experience*, a compact memory built by executing both the LLM and the agent on a shared seed set without knowing the ground truth, as shown in Figure 2. Rather than serving as supervision or calibration data, this early experience acts as a lightweight behavioral reference that exposes systematic differences between the two systems.

To systematically study this routing problem, we construct **RouteBench**, a benchmark specifically designed to evaluate the decision boundaries of LLMs to route between LLMs and agents. Unlike conventional evaluation suites that assume a static distribution, RouteBench assesses routing generalization across three progressively challenging dimensions: standard in-domain tasks, linguistically perturbed queries for robustness, and out-of-domain scenarios. This design enables a rigorous assessment of how well routers can balance performance and cost when facing both familiar and novel task distributions.

Finally, we evaluate BoundaryRouter on Routebench, which reduces average inference time by **60.6**% compared to the agent and achieves **28.6**% performance improvement over direct LLM inference, demonstrating a clearly better cost–performance trade-off. Using BoundaryRouter, we further evaluate 14 contemporary models on RouteBench. Among frontier models, GPT-5, Gemini-3-Pro-Preview, and Gemini-2.5-Pro achieve the strongest overall routing performance. Compared with simple prompt



(a) Average RouteBenchScore across models. (b) Overall Routing Effectiveness of BoundaryRouter.

Figure 3: **Overall routing performance and cost trade-offs on RouteBench**.(a) Average RouteBenchScore across all evaluation sets for different models, sorted in descending order; (b) Comparison of routing effectiveness across different routing strategies. The bar chart reports the average routing score on RouteBench for basic prompt-based routing, retrieval-based (RAG) routing, and our routing method across three backbone models. Our method consistently achieves higher routing performance than both baselines, demonstrating its effectiveness as a general routing strategy.

routing or retrieval-augmented generation (RAG) routing, BoundaryRouter improves routing quality by **37.9**% and **8.2**%, respectively, confirming the effectiveness of early experience and rubric-guided reasoning. Our findings highlight that in cold-start settings without routing labels or ground-truth, early behavioral signals paired with rubric-constrained reasoning can enable reliable routing between LLMs and agents, offering a practical path toward scalable coordination in heterogeneous reasoning systems.

2 Related Works

Routing in LLM, LLM Agents and Multi-Model Systems. Task routing has become increasingly central to scaling language model systems, especially as workloads grow in diversity and cost sensitivity. Early efforts established the foundation by evaluating routing performance across diverse benchmark datasets (Shnitzer et al., 2023). A series of recent works explore how to dynamically dispatch queries across multiple models, agents, or tools to optimize utility through universal routing frameworks (Jitkrittum et al., 2025). Some approaches, like Router-R1 (Zhang et al., 2025a), integrate routing into multi-step inference, where models iteratively decide which component to consult based on intermediate reasoning signals, often using reinforcement learning to balance accuracy and cost. Others operate in multi-agent setups, coordinating agents with different roles or specializations via hierarchical planning, graph-based dispatching, or role-aware context filtering (Yue et al., 2025; Zhang et al., 2025d; Liu et al., 2025b). Routing under resource constraints has also received attention, with methods selecting models adaptively based on utility-cost trade-offs (Panda et al., 2025), lookahead mechanisms (Huang et al., 2025), or test-time compute optimization (Ding et al., 2025a). Furthermore, reward-guided ensembles have been proposed to route queries to the most capable expert model (Lu et al., 2024). In parallel, retrieval-augmented reasoning systems treat routing as a step-wise selection over knowledge bases or tools (Peng et al., 2025). While these directions reflect growing interest in adaptive coordination, they typically focus on routing within homogeneous spaces, across models, agents, or tools, but not across them. Our work fills this gap by studying routing between LLMs and agents, enabling task-level decisions that exploit their complementary strengths.

Learning from Early Experience and Self-Evolving Agents. To build more adaptive and autonomous systems, recent work has explored how agents can learn from their early history. Reflexion Shinn et al. (2023) and Voyager Wang et al. (2023) demonstrate that agents can reflect on failures, consolidate long-term memory, and develop reusable skills through language-based feedback and exploration. Beyond retrospective improvement, some methods adapt agents at test time by detecting errors or uncertainty and updating internal components accordingly Acikgoz et al. (2025), while others leverage early interaction data to bootstrap policies via future-consistent behavior modeling

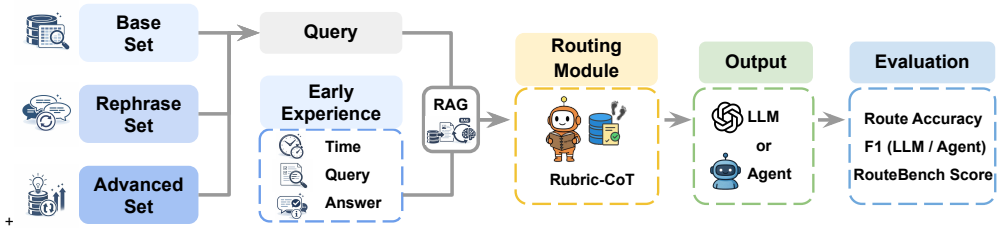


Figure 4: Overview of our routing pipeline. A query from RouteBench is routed using early-experience retrieval-augmented (RAG) and a CoT-based routing module, which delegates the task to either an LLM or an Agent. Performance is evaluated on RouteBench via routing accuracy and per-solver F1, aggregated into a RouteBenchScore.

Zhang et al. (2025c). These ideas have also reshaped how agents are architected. Instead of relying on static pipelines, agents can evolve dynamically by refining their internal logic, memory, and prompting strategies over time Wu et al. (2025). A recent survey consolidates these directions into the emerging framework of self-evolving agents, which emphasizes the shift from static models to continually adapting, self-refining systems (Gao et al., 2025). We extend experience-based learning from task execution to routing, enabling agents to improve delegation decisions across LLMs and agents, a dimension rarely addressed in prior work.

Reasoning-Enhanced Decision Making. Chain-of-Thought (CoT) prompting, introduced by Wei et al. (Wei et al., 2022), enables large language models to reason through intermediate steps before producing final answers. Beyond accuracy gains, CoT supports decision-making within models and agents. In ReAct (Yao et al., 2022), reasoning traces guide tool use and subroutine selection, while other work employs CoT for task decomposition and option evaluation (Zhou et al., 2022; Kojima et al., 2022). Recent advances incorporate rubric-guided CoT, where reasoning is shaped by explicit evaluation criteria rather than implicit preferences. This approach improves consistency and alignment in tasks such as text generation, code evaluation, and geospatial planning by ensuring that reasoning respects domain-specific constraints (Pathak et al., 2025; Chen et al., 2025). Building on this direction, we apply CoT to agent routing, helping models reason explicitly, under rubric guidance, about which agent or submodel is best suited for a given task. This reframes CoT as a structured coordination mechanism that enhances routing transparency and reliability.

3 Learning Agent Routing from Early Experience

3.1 Routing Module Overview

The routing module is an LLM-based decision system that routes each incoming query to either a lightweight LLM (low latency and token cost) or a full agent (higher cost but stronger tool-augmented reasoning). Concretely, the router is a *pluggable* routing LLM that conditions on (i) the input query and (ii) retrieved early-experience cases containing solver outputs and runtime. Given these signals, the router follows a rubric-guided reasoning (Box 4.2) to trade off expected answer quality against latency and then outputs the routing decision. Formally, given an input query x, the router produces $\text{Route}(x) \in \{\text{LLM}, \text{Agent}\}$. Importantly, the routing module is **training-free**: it uses no gradient updates and does not require supervision on routing labels. This design makes the router easy to deploy and to adapt to new tasks or new agent implementations by updating only the early-experience memory and retrieval components.

3.2 Learning from Early Experience

A central difficulty in LLM–agent routing is *cold start*: when the router is deployed, we typically do not have ground-truth for the incoming queries and therefore cannot obtain reliable routing labels. This makes standard supervised router training impractical. To address this, we introduce **Learning from Early Experience**, which provides the router with a compact memory of observable solver behavior, with the pipeline overview shown in Fig. 4.

Constructing the early-experience memory. We first sample a small seed set of questions $\mathcal{D}_{\text{seed}}$ and run both candidate solvers—a lightweight LLM and a full agent—on the same inputs. For each $x \in \mathcal{D}_{\text{seed}}$, we record only deployment-time observable information:

- the question x,
- the LLM output y^{LLM} and latency t^{LLM},
- the agent output y^{Agent} and latency t^{Agent}.

Crucially, we do not store gold answers, correctness labels, or rewards. The resulting memory

$$\mathcal{M} = \{(x_i, y_i^{\text{LLM}}, y_i^{\text{Agent}}, t_i^{\text{LLM}}, t_i^{\text{Agent}})\}_{i=1}^{N}$$

captures systematic differences in the two systems' behavior (e.g., response and runtime) without requiring supervision.

Retrieval-augmented routing. At inference time, given a new query x, we retrieve the top-K most similar records from $\mathcal{M}$ using a hybrid retriever (sparse lexical matching plus dense semantic similarity):

$$\text{Retrieve}(\mathcal{M}, x) = \{(x_k, y_k^{\text{LLM}}, y_k^{\text{Agent}}, t_k^{\text{LLM}}, t_k^{\text{Agent}})\}_{k=1}^{K}.$$

These retrieved cases are provided to the routing LLM as evidence. The router compares the current query against the retrieved questions and inspects the two solvers' outputs and latencies to infer regularities, e.g., whether similar questions previously led the agent to produce slower, multi-step reasoning, or how similarity in phrasing or structure correlates with the relative efficiency and behavior of the two systems.

3.3 Rubric-guided Chain-of-Thought Routing

In the cold-start setting, the router must make a decision without access to ground-truth answers or routing labels. A natural approach to routing is to allow the routing LLM to reason explicitly before selecting between the LLM and the agent. Chain-of-Thought (CoT) prompting has been shown to improve structured decision-making by encouraging step-by-step reasoning (Wei et al., 2022; Yao et al., 2022; Kojima et al., 2022). However, in our setting, routing is not an open-ended reasoning task: decisions should follow explicit behavioral criteria, such as comparing answer characteristics and response times observed in early experience, as formalized in Box 4.2.

Direct, free-form CoT does not guarantee that the routing LLM will consistently attend to these criteria, especially under paraphrasing or distribution shift. To align the reasoning process with the rule-based nature of routing, we therefore adopt a rubric-guided CoT formulation that explicitly encodes the evaluation protocol into the prompt. As shown in Fig. 7, the router is required to follow a fixed decision rubric that reflects the actual dimensions available in the early-experience memory, rather than relying on unconstrained reasoning.

4 RouteBench: Benchmarking LLM Routing between LLM and Agent

Routing between heterogeneous reasoning systems requires a benchmark with diverse tasks, clear supervision, and controlled distribution shifts. To address this, we introduce **RouteBench**, a benchmark designed to evaluate how effectively a model assigns queries between a lightweight LLM and a full agent. RouteBench consists of a curated question pool drawn from GAIA and MMLU, paired solver outputs from both systems, and human-annotated routing labels. Each instance includes the question, solver predictions, latency, and the ground-truth routing decision. To assess routing generalization, RouteBench provides three evaluation sets covering in-domain, paraphrased, and out-of-domain conditions. Routing performance is evaluated at both the instance level and the set level, using routing accuracy, solver-specific F1, and the final RouteBenchScore (Fig. 5).

4.1 Benchmark Curation

To form the base question pool, we sample from two established sources. GAIA (Mialon et al., 2023) provides open-ended reasoning tasks that reflect real-world problem solving. MMLU (Hendrycks

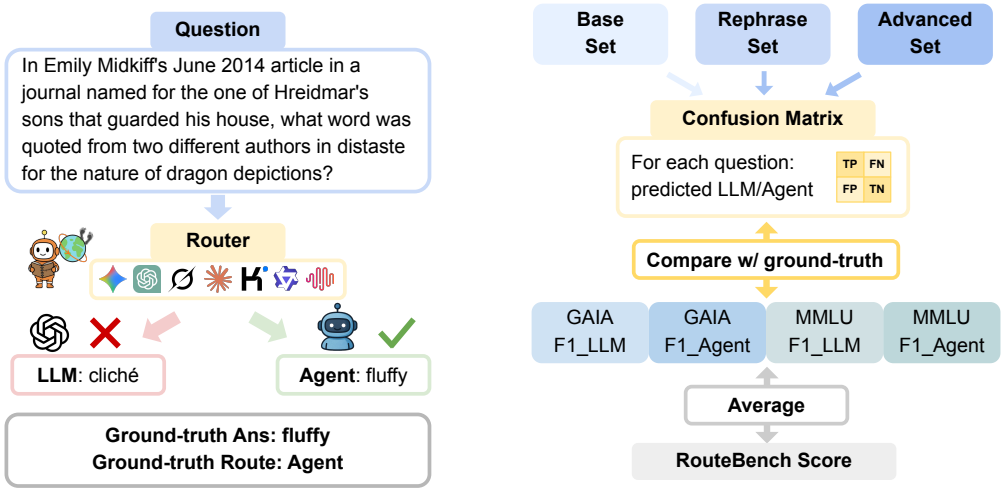


Figure 5: Overview of the RouteBench evaluation framework. On the left is the Single-instance routing process: solver outputs determine the ground-truth route, and the router's decision is evaluated via exact-match accuracy. On the right is Set-level evaluation: the router's predictions over all questions yield per-solver F1 scores (LLM vs. Agent), which are averaged to produce the final RouteBenchScore.

et al., 2021) spans 57 academic subjects and provides structured knowledge questions. We randomly sample **30 GAIA questions** and **57 MMLU questions** (one per subject), producing a compact yet diverse collection that covers factual recall, symbolic reasoning, multi-step planning, and open-domain inference.

For each selected question, we collect solver predictions from two systems: an LLM and a full agent with tool-use and intermediate reasoning capabilities. All samples undergo manual review to ensure semantic clarity, correctness of solver traces, and consistency of formatting. This curated pool serves as the foundation for all evaluation splits introduced in later sections.

4.2 Benchmark Composition

Each RouteBench instance can be viewed as a 5-tuple

$$(x, y^{\text{LLM}}, y^{\text{Agent}}, y^*, d),$$

where these elements correspond to the question, LLM prediction, agent prediction, ground-truth answer, and routing decision, respectively. The first four fields describe the task and solver behaviors. The final field, d, is the human-annotated label indicating which solver provides the preferred answer for that question.

The label of d follows a fixed deterministic rule:

Ground-truth Routing Rule

1. Correctness priority: If only one solver produces the correct answer, choose that solver.
2. Efficiency tie-break: If both solvers are correct, choose the one with shorter response time.
3. Failure fallback: If both solvers are incorrect, choose the agent, which has a higher chance of recovery through multi-step reasoning.

This labeling scheme ensures that RouteBench measures the ability to infer better solver selection, rather than answer-generation accuracy.

4.3 Evaluation Sets for Routing Generalization

From the curated question pool, we construct three evaluation sets designed to assess routing under different generalization conditions.

Base Set (In-domain). The Base Set contains 30 GAIA and 57 MMLU questions with solver predictions and routing labels. It reflects the in-domain distribution and also serves as the early-experience corpus for retrieval-augmented routing. All questions are evaluated in their original form.

Rephrase Set (Paraphrased In-domain). The Rephrase Set is created by paraphrasing each Base Set question using a controlled LLM-based rewriting process that preserves semantics while modifying surface form. This set evaluates routing stability under linguistic variation without changing task content.

Advanced Set (Out-of-domain). It consists of GAIA and MMLU questions that are disjoint from the Base and Rephrase Sets. Although drawn from the same benchmarks, these questions differ in topic and reasoning structure, providing a controlled out-of-domain evaluation.

4.4 Evaluation Metrics

RouteBench uses a small set of complementary metrics to evaluate routing and provide a single scalar score for model comparison. Instance-level accuracy and solver-level PRF metrics serve as diagnostic measures, while RouteBenchScore is the primary metric for overall routing quality.

Instance-level routing accuracy. For each question, the routing model outputs a binary decision indicating which solver to use. We compute exact-match accuracy by comparing predictions against the human-annotated ground-truth routing labels defined in 4.2. This metric reflects overall decision correctness but does not provide detailed information about each solver's routing characteristics.

Solver-level PRF metrics. To characterize routing behavior for each solver, we compute Precision, Recall, and F1 for the LLM and the agent separately. Let class A denote routing to the LLM and class B denote routing to the agent. For each class, RouteBench provides the number of ground-truth assignments, tot_A and tot_B, while a routing model yields true positives, tp_A and tp_B. False positives and false negatives follow by symmetry, $\text{fp}_A = \text{fn}_B$ and $\text{fp}_B = \text{fn}_A$. Precision, Recall, and F1 are then computed independently for each solver within each evaluation set.

Final RouteBench score. To obtain a single summary metric, we average the solver-level F1 scores across both solvers and task sources:

$$\text{RouteBenchScore} = \frac{1}{4} \sum_{s} \sum_{d} \text{F1}_s^d.$$

Here, s ranges over the two solvers (LLM and agent), and d ranges over the two task sources (GAIA and MMLU). This score reflects how consistently a routing model selects the correct solver across both open-domain and academic reasoning tasks, while weighting the two solvers and the two benchmarks equally.

Primary comparison metric. Although instance-level accuracy and solver-level PRF metrics are useful for diagnostic analysis, **RouteBenchScore is the primary metric used for comparing routing models**. All model-to-model comparisons in this paper are based on this score, as it provides a concise summary of overall routing effectiveness across evaluation settings.

5 Experiment

5.1 Experiment Setup

Models. To construct the early-experience database used for retrieval-augmented routing, we adopt GPT-4o Hurst et al. (2024) as the representative LLM. For the agent implementation, we use Claude Sonnet 4 for agent logic and GPT-4o as the tool-calling backbone within the SmolAgent Open DeepResearch framework Roucher et al. (2025). We evaluate routing performance across a broad set of contemporary large language models, covering both state-of-the-art systems and widely deployed alternatives. The evaluated models include GPT-5 OpenAI (2025a), GPT-5.2 OpenAI (2025b), GPT-5-nano OpenAI (2025a), Gemini-3-Pro-Preview Google DeepMind, Gemini-3-Flash-Preview Doshi & Gemini Team (2025), Gemini-2.5-Pro Comanici et al. (2025), Gemini-2.5-flash Comanici et al. (2025), Claude Sonnet 4 Anthropic (2025a), Claude Sonnet 4.5 Anthropic (2025b), MinMax-M2 MiniMax

Table 1: **Routing performance and inference cost on RouteBench**. Results are reported for LLM-only, Agent-only, and our routing method on the Base, Rephrase, and Advanced evaluation sets. For each set, we report routing accuracy (Acc.) and average inference time (Time, in seconds) on MMLU, GAIA, and their average (Avg). Our routing method consistently achieves higher accuracy than the LLM-only baseline while substantially reducing inference cost compared to the Agent-only baseline across all settings, striking a balance between the performance and cost.

Model	Base Set						Rephrase Set						Advanced Set					
	MMLU		GAIA		Avg		MMLU		GAIA		Avg		MMLU		GAIA		Avg	
	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time
LLM	0.754	2.22	0.1	8.633	0.528	4.431	0.754	0.533	0.13	5.58	0.54	2.27	0.860	4.17	0.23	5.23	0.64	4.53
Agent	0.895	174.67	0.533	436.60	0.77	264.99	0.895	112.16	0.6	279.80	0.793	169.70	0.982	147.32	0.667	393.428	0.87	232.18
BoundaryRouter	0.8772	26.6	0.4	244.86	0.713	101.86	0.842	24.35	0.467	222.89	0.713	92.81	0.877	36.08	0.567	197.03	0.77	91.58

AI (2025), Qwen3-32b Yang et al. (2025), Grok-4 xAI (2025), Kimi-K2-Thinking Moonshot AI (2025), DeepSeek-v3.2 Liu et al. (2025a).

Evaluation Details. All experiments are conducted on RouteBench. For each question, the resulting decision is compared against the ground-truth routing decision, d. Performance is computed using the official RouteBench scoring procedure described in Section 4.4. In all experiments, the same BoundaryRouter framework is used, with different LLMs adopted as the routing model. All API-based models are evaluated using the default parameters provided by Openrouter.

5.2 Main Results

Table 1 summarizes the accuracy-latency trade-off on the three splits (Base, Rephrase, Advanced), reported separately on MMLU and GAIA, and averaged across two sources.

Performance of BoundaryRouter. Across all three evaluation sets, BoundaryRouter consistently achieves a favorable balance between accuracy and inference cost. Compared to the LLM-only run, BoundaryRouter substantially improves accuracy on both MMLU and GAIA, with an average relative improvement of 28.6%. Compared to Agent-only routing, our method reduces inference time by an order of magnitude, with a **60.6**% relative reduction, while retaining a large fraction of the agent's accuracy, only a relative 11.5% decrease. On the Base Set, the LLM-only baseline exhibits low GAIA accuracy (0.10) despite fast inference, whereas the Agent achieves higher accuracy at a prohibitive average cost of 264.99 seconds. Our routing method bridges this gap, improving average accuracy to 0.713 while reducing inference time to 101.86 seconds. A similar trend is observed on the Rephrase Set. Crucially, in the out-of-domain set, the Advanced Set, BoundaryRouter still performs well. It achieves a strong accuracy of 0.77 with less than half of the Agent's inference time. This result indicates that the routing strategy generalizes beyond the in-domain distribution and remains effective under distribution shift. These results show that BoundaryRouter effectively routes easy questions to the LLM and leaves hard questions for the Agent. This selective invocation allows the system to maintain strong performance while avoiding the high cost of invoking the agent for every query.

Comparison with LLM-only and Agent-only baselines. Overall, while the Agent achieves 43.7% higher accuracy than the LLM baseline, it is nearly **60**$\times$ slower in inference time. Neither baseline alone provides a satisfactory balance between accuracy and inference cost. The LLM baseline offers low latency but limited performance on GAIA, while the Agent baseline improves performance at the cost of extremely high inference time. BoundaryRouter bridges this gap by combining the strengths of both approaches, achieving strong accuracy while maintaining a much lower inference cost.

5.3 RouteBench Results

Overall routing performance. Table 2 reports routing performance across 14 models on RouteBench. Models are ranked by their overall average score. We observe a clear performance stratification across model families. **GPT-5** achieves the strongest overall routing performance, with an average score of 0.75, followed closely by **Gemini-3-Pro-Preview** (0.734) and **Gemini-2.5-Pro** (0.726). These models remain stable even when questions are rewritten or shifted to new topics, suggesting that their routing patterns are less sensitive to changes in surface form or content. These top-ranked models consistently exhibit strong solver discrimination on both MMLU and GAIA, indicating robust routing decisions across domains and distribution shifts. This finding also aligns with these models'

Table 2: **Model Performance on RouteBench**. Models are ranked from top to bottom by their overall average score. All rankings are computed using the full-precision underlying scores, while values reported in the table are rounded to two decimal places for readability, but rounded to three decimal places for the overall average for easier comparison.

Model	Avg.	Base Set					Rephrase Set					Advanced Set				
		MMLU		GAIA		Avg.	MMLU		GAIA		Avg.	MMLU		GAIA		Avg.
		LLM	Agent	LLM	Agent		LLM	Agent	LLM	Agent		LLM	Agent	LLM	Agent	
GPT-5	0.750	0.95	0.86	0.60	0.92	<u>0.83</u>	0.95	0.86	0.55	0.90	**0.81**	0.81	0.32	0.46	0.85	<u>0.61</u>
Gemini-3-Pro-Preview	0.734	0.94	0.83	0.55	0.90	0.80	0.95	0.86	0.44	0.87	<u>0.78</u>	0.79	0.24	0.55	0.90	**0.62**
Gemini-2.5-Pro	0.726	0.93	0.80	0.67	0.94	**0.84**	0.93	0.80	0.36	0.86	0.75	0.70	0.38	0.44	0.90	<u>0.61</u>
Gemini-3-Flash-Preview	0.716	0.95	0.86	0.50	0.88	0.80	0.94	0.83	0.44	0.87	0.77	0.76	0.22	0.44	0.90	0.58
Grok-4	0.685	0.95	0.86	0.44	0.90	0.79	0.95	0.86	0.43	0.83	0.77	0.73	0.31	0.17	0.79	0.50
Gemini-2.5-flash	0.682	0.95	0.86	0.50	0.88	0.80	0.93	0.80	0.36	0.86	0.74	0.74	0.15	0.33	0.83	0.51
Qwen3-32b	0.667	0.97	0.89	0.35	0.74	0.74	0.95	0.85	0.27	0.76	0.71	0.81	0.26	0.42	0.73	0.56
DeepSeek-v3.2	0.667	0.92	0.76	0.38	0.78	0.71	0.94	0.83	0.46	0.80	0.76	0.84	0.29	0.27	0.76	0.54
GPT-5.2	0.656	0.93	0.79	0.38	0.77	0.72	0.94	0.80	0.46	0.80	0.75	0.76	0.08	0.40	0.80	0.51
Kimi-K2-Thinking	0.651	0.93	0.77	0.40	0.80	0.73	0.92	0.76	0.44	0.76	0.72	0.80	0.18	0.33	0.71	0.51
Claude-4.5-sonnet	0.635	0.95	0.85	0.33	0.71	0.71	0.95	0.85	0.68	0.32	0.70	0.85	0.22	0.29	0.62	0.49
MinMax-M2	0.630	0.86	0.60	0.35	0.74	0.64	0.90	0.73	0.25	0.73	0.65	0.75	0.32	0.50	0.82	0.60
Claude-4-sonnet	0.615	0.93	0.77	0.27	0.76	0.68	0.95	0.78	0.33	0.71	0.69	0.84	0.00	0.33	0.71	0.47
GPT-5-nano	0.578	0.85	0.63	0.35	0.74	0.64	0.87	0.65	0.22	0.66	0.60	0.78	0.34	0.32	0.51	0.49

Table 3: Comparison of the three routing variants across GPT-5, Gemini 2.5 Pro, and Claude Sonnet 4 on Base, Rephrase, and Advanced Set. The table shows that early-experience memory and structured reasoning together provide the strongest and most stable performance.

	GPT-5				**Gemini-2.5-Pro**				**Claude-4-Sonnet**			
	Base	Rephrase	Advanced	Avg.	Base	Rephrase	Advanced	Avg.	Base	Rephrase	Advanced	Avg.
Prompt Routing	0.48	0.48	0.26	0.41	0.55	0.57	0.58	0.57	0.52	0.57	0.56	0.55
RAG Routing	0.80	0.80	0.54	0.72	0.76	0.77	0.42	0.65	0.61	0.60	0.51	0.58
BoundaryRouter	0.83	0.81	0.61	0.75	0.78	0.79	0.54	0.71	0.73	0.65	0.56	0.65

abilities on other benchmarks, like AIME25 [1], LiveCodeBench Jain et al. (2024), and Humanity's last exam Phan et al. (2025).

Routing is surprisingly still a relatively hard problem. Even in the in-domain setting where questions closely match the early experience and the task is binary, routing accuracy remains far from perfect across all models. This difficulty becomes more pronounced under distribution shift, but its presence in the Base set already indicates that effective routing is not a solved problem, even without paraphrasing or topic change.

Advanced Set Drives Ranking Separation. While routing performance is similar across models on the Base and Rephrase sets, differences become large on the Advanced set, where top models retain average scores around 0.61–0.62, mid-tier models fall to 0.54–0.56, and lower-ranked models drop below 0.50. This ranking reshuffle suggests that out-of-domain routing ability, rather than in-domain decision matching, is the primary factor distinguishing strong routing models from weaker ones.

5.4 Ablation Study

To understand the contribution of each component in our routing framework, Table 3 compares three variants: basic Prompt Routing, RAG Routing, and our Rucirc-guided CoT Routing with early experience (i.e., RAG), across GPT-5, Gemini 2.5 Pro, and Claude Sonnet 4. The corresponding prompts for the two ablation baselines are provided in Appendix A.4.

Prompt Routing. This variant removes early-experience memory entirely. It selects between the LLM and the agent using only high-level capability profiles provided in the prompt, refer to Appendix A.4. Without early experience, routing decisions depend strongly on surface cues, leading to unstable behavior under paraphrasing and distribution shift. Even in the Base set, average scores remain low (e.g., 0.41 for GPT-5 and 0.55 for Claude-4-Sonnet), indicating that capability-aware reasoning is insufficient for reliable solver selection.

[1] https://huggingface.co/datasets/yentinglin/aime_2025

RAG Routing. This variant introduces early-experience memory but removes rubric-guided reasoning. Retrieved behavioral examples are shown to the router, which must directly output a binary routing decision without reasoning (prompt in Appendix A.4). It substantially improves performance, with an average of 27.5% improvement, especially on the Base and Rephrase sets by introducing early-experience retrieval, suggesting that access to historical solver behavior provides useful signals for routing, even without the gold answer.

Rubric-Guided CoT with Early Experience (BoundaryRouter). It combines early-experience memory with rubric-guided chain-of-thought reasoning, consistently achieves the best performance across all models, and sets, with an increase of 37.9% compared to Prompt Routing and of 8.2% compared to RAG Routing. Notably, it yields the highest Advanced-set scores for all three LLMs, also maintaining strong in-domain performance. This demonstrates that structured reasoning is essential for effectively interpreting retrieved experiences and making stable routing decisions under the distribution shift.

6 Discussion and Conclusion

We study cold-start routing between direct lightweight LLM inference and full agentic execution, where ground-truth labels are unavailable at deployment time. To address this, we propose BoundaryRouter, a training-free router that learns from early experience, and introduce RouteBench, a benchmark for evaluating LLM–agent routing under in-domain, paraphrased, and out-of-domain settings. Experiments show that BoundaryRouter improves the accuracy–latency trade-off over baselines and remains robust under paraphrasing and distribution shift. These results suggest that early experience provides useful signals for routing decisions even without access to ground-truth answers, and that structured reasoning helps maintain stable decisions when task distributions change. While our current framework focuses on binary routing between an LLM and a single agent pipeline, future work may explore more complex routing scenarios involving multiple agents or heterogeneous tools. Overall, our findings highlight routing as an important component for improving the efficiency and scalability of hybrid LLM–agent systems.

REPRODUCIBILITY STATEMENT

We provide sufficient details to enable full reproduction of RouteBench and the BoundaryRouter framework. RouteBench is constructed from publicly available GAIA and MMLU benchmarks with explicitly defined sampling, annotation, and evaluation procedures. Each instance includes solver outputs, latency, and deterministic routing labels based on a fixed decision rule. BoundaryRouter is training-free and relies on early-experience memory, retrieval, and rubric-guided routing, all of which are fully specified in the paper and appendix, including prompts and decision protocols. All models are evaluated using the same splits and scoring procedure, including routing accuracy and RouteBenchScore. We will release the RouteBench dataset, routing prompts, and evaluation code upon publication to ensure full reproducibility.

REFERENCES

Emre Can Acikgoz, Cheng Qian, Heng Ji, Dilek Hakkani-Tür, and Gokhan Tur. Self-improving llm agents at test-time. *arXiv preprint arXiv:2510.07841*, 2025.

Anthropic. Introducing claude 4, May 2025a. URL https://www.anthropic.com/news/claude-4.

Anthropic. Claude sonnet 4.5. Anthropic Model Release, September 2025b. URL https://www.anthropic.com/claude/sonnet. Released September 29, 2025.

Yiheng Chen, Lingyao Li, Zihui Ma, Qikai Hu, Yilun Zhu, Min Deng, and Runlong Yu. Empowering llm agents with geospatial awareness: Toward grounded reasoning for wildfire response. *arXiv preprint arXiv:2510.12061*, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian Del Carmen Hipolito Garcia, Menglin Xia, Laks VS Lakshmanan, Qingyun Wu, and Victor Rühle. Best-route: Adaptive llm routing with test-time optimal compute. *arXiv preprint arXiv:2506.22716*, 2025a.

Keyan Ding, Jing Yu, Junjie Huang, Yuchen Yang, Qiang Zhang, and Huajun Chen. Scitoolagent: a knowledge-graph-driven scientific agent for multitool integration. *Nature Computational Science*, pp. 1–11, 2025b.

Tulsee Doshi and Gemini Team. Gemini 3 flash: Frontier intelligence built for speed. Google Product Blog, December 2025. URL https://blog.google/products-and-platforms/products/gemini/gemini-3-flash/. Posted December 17, 2025.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, et al. A survey of self-evolving agents: On path to artificial super intelligence. *arXiv preprint arXiv:2507.21046*, 2025.

Google DeepMind. Gemini 3 pro. Web page. URL https://deepmind.google/models/gemini/pro/. Accessed 2026-01-28.

H2O.ai. H2OGPT Generative AI Platform. Web Application, 2025. URL https://h2ogpte.genai.h2o.ai/.

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

Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Bowei Xia, Tao Sun, Ziyu Ye, Zhaoxuan Jin, Yingru Li, Qiguang Chen, et al. Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation. *arXiv preprint arXiv:2505.23885*, 2025.

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

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

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

Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Jeevesh Juneja, Congchao Wang, Zifeng Wang, Alec Go, Chen-Yu Lee, Pradeep Shenoy, Rina Panigrahy, et al. Universal model routing for efficient llm inference. *arXiv preprint arXiv:2502.08773*, 2025.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213, 2022.

Jierui Li, Hung Le, Yingbo Zhou, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Codetree: Agent-guided tree search for code generation with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3711–3726, 2025.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025a.

Jun Liu, Zhenglun Kong, Changdi Yang, Fan Yang, Tianqi Li, Peiyan Dong, Joannah Nanjekye, Hao Tang, Geng Yuan, Wei Niu, et al. Rcr-router: Efficient role-aware context routing for multi-agent llm systems with structured memory. *arXiv preprint arXiv:2508.04903*, 2025b.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. Routing to the expert: Efficient reward-guided ensemble of large language models. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1964–1974, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.109. URL https://aclanthology.org/2024.naacl-long.109/.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*, 2023.

MiniMax AI. Minimax-m2: A compact moe model for coding and agentic workflows. Open-source model, 2025. URL https://huggingface.co/MiniMaxAI/MiniMax-M2. Model weights and documentation available at Hugging Face and GitHub.

Moonshot AI. Introducing kimi k2 thinking. Web page, November 2025. URL https://moonshotai.github.io/Kimi-K2/thinking.html. Accessed: 2026-01-28.

OpenAI. Introducing GPT-5. https://openai.com/index/introducing-gpt-5/, August 2025a. Accessed: 2026-01-27.

OpenAI. Introducing gpt-5.2, December 2025b. URL https://openai.com/index/introducing-gpt-5-2/. Product release blog post.

OpenRouter. Web search | add real-time web data to ai model responses. https://openrouter.ai/docs/features/web-search, 2025. Accessed: 2025-11-19.

Pranoy Panda, Raghav Magazine, Chaitanya Devaguptapu, Sho Takemori, and Vishal Sharma. Adaptive llm routing under budget constraints. *arXiv preprint arXiv:2508.21141*, 2025.

Aditya Pathak, Rachit Gandhi, Vaibhav Uttam, Arnav Ramamoorthy, Pratyush Ghosh, Aaryan Raj Jindal, Shreyash Verma, Aditya Mittal, Aashna Ased, Chirag Khatri, et al. Rubric is all you need: Improving llm-based code evaluation with question-specific rubrics. In *Proceedings of the 2025 ACM Conference on International Computing Education Research V. 1*, pp. 181–195, 2025.

Chunyi Peng, Zhipeng Xu, Zhenghao Liu, Yishan Li, Yukun Yan, Shuo Wang, Zhiyuan Liu, Yu Gu, Minghe Yu, Ge Yu, et al. Learning to route queries across knowledge bases for step-wise retrieval-augmented reasoning. *arXiv preprint arXiv:2505.22095*, 2025.

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

Jiahao Qiu, Xinzhe Juan, Yimin Wang, Ling Yang, Xuan Qi, Tongcheng Zhang, Jiacheng Guo, Yifu Lu, Zixin Yao, Hongru Wang, et al. Agentdistill: Training-free agent distillation with generalizable mcp boxes. *arXiv preprint arXiv:2506.14728*, 2025a.

Jiahao Qiu, Xuan Qi, Hongru Wang, Xinzhe Juan, Yimin Wang, Zelin Zhao, Jiayi Geng, Jiacheng Guo, Peihang Li, Jingzhe Shi, et al. Alita-g: Self-evolving generative agent for agent generation. *arXiv preprint arXiv:2510.23601*, 2025b.

Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, et al. Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution. *arXiv preprint arXiv:2505.20286*, 2025c.

Jiahao Qiu, Fulian Xiao, Yimin Wang, Yuchen Mao, Yijia Chen, Xinzhe Juan, Shu Zhang, Siran Wang, Xuan Qi, Tongcheng Zhang, et al. On path to multimodal historical reasoning: Histbench and histagent. *arXiv preprint arXiv:2505.20246*, 2025d.

Aymeric Roucher, Albert Villanova del Moral, Thomas Wolf, Leandro von Werra, and Erik Kaunismäki. 'smolagents': a smol library to build great agentic systems. `https://github.com/huggingface/smolagents`, 2025.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652, 2023.

Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. Large language model routing with benchmark datasets. *arXiv preprint arXiv:2309.15789*, 2023.

MiroMind AI Team. Miroflow: A high-performance open-source research agent framework. `https://github.com/MiroMindAI/MiroFlow`, 2025.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Zhizheng Wang, Qiao Jin, Chih-Hsuan Wei, Shubo Tian, Po-Ting Lai, Qingqing Zhu, Chi-Ping Day, Christina Ross, Robert Leaman, and Zhiyong Lu. Geneagent: self-verification language agent for gene-set analysis using domain databases. *Nature Methods*, pp. 1–9, 2025.

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

Rong Wu, Xiaoman Wang, Jianbiao Mei, Pinlong Cai, Daocheng Fu, Cheng Yang, Licheng Wen, Xuemeng Yang, Yufan Shen, Yuxin Wang, et al. Evolver: Self-evolving llm agents through an experience-driven lifecycle. *arXiv preprint arXiv:2510.16079*, 2025.

xAI. Grok 4. xAI Model Announcement, July 2025. URL `https://x.ai/news/grok-4`. Posted July 9, 2025.

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

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik R Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2405.15793.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

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

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

Hongwei Zhang, Ji Lu, Shiqing Jiang, Chenxiang Zhu, Li Xie, Chen Zhong, Haoran Chen, Yurui Zhu, Yongsheng Du, Yanqin Gao, et al. Co-sight: Enhancing llm-based agents via conflict-aware meta-verification and trustworthy reasoning with structured facts. *arXiv preprint arXiv:2510.21557*, 2025b.

Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, et al. Agent learning via early experience. *arXiv preprint arXiv:2510.08558*, 2025c.

Zheyuan Zhang, Kaiwen Shi, Zhengqing Yuan, Zehong Wang, Tianyi Ma, Keerthiram Murugesan, Vincent Galassi, Chuxu Zhang, and Yanfang Ye. Agentrouter: A knowledge-graph-guided llm router for collaborative multi-agent question answering. *arXiv preprint arXiv:2510.05445*, 2025d.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

Huichi Zhou, Yihang Chen, Siyuan Guo, Xue Yan, Kin Hei Lee, Zihan Wang, Ka Yiu Lee, Guchun Zhang, Kun Shao, Linyi Yang, et al. Agentfly: Fine-tuning llm agents without fine-tuning llms. *arXiv preprint arXiv:2508.16153*, 2025.

A Appendix

A.1 Use of LLMs

LLMs were used solely to improve the clarity and readability of the manuscript.

A.2 Example

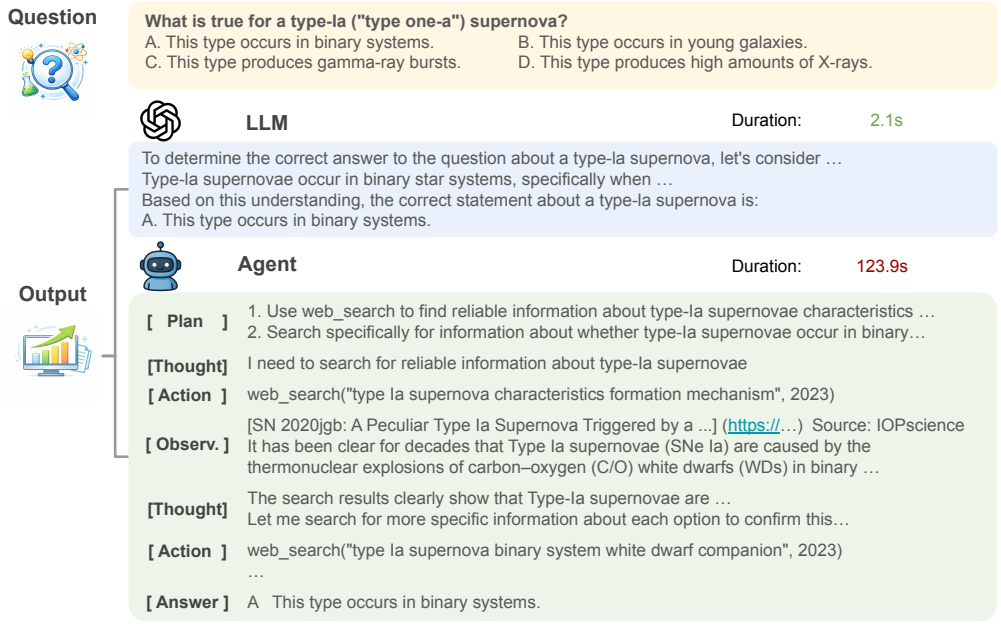


Figure 6: Example illustrating routing between direct LLM inference and agent execution. For this factual multiple-choice question, the LLM produces a correct answer quickly (2.1s), while the agent follows a multi-step search-and-reasoning process that is substantially slower (123.9s) but arrives at the same conclusion. This example highlights the accuracy–latency trade-off that motivates routing, where many queries fall within the capability boundary of direct LLM inference and do not require full agent execution.

A.3 METHOD DETAILS

Regular CoT (Unstructured)

```
You are an intelligent routing system that determines which model
should answer a question.
Original question: {original_question}
Retrieved similar question examples:
{retrieved examples}
Let's think step by step about which model should answer the
original question based on the historical data above.
Report your thoughts, and must finish your answer with the following
template:
FINAL ANSWER: [YOUR FINAL ANSWER].
YOUR FINAL ANSWER should be either "YES" (use Agent) or "NO" (use
LLM).
```

Rubric-guided CoT (Structured)

```
You are an intelligent routing system that determines which model
should answer a question.
Original question: {original_question}
Retrieved similar question examples:
{retrieved examples}
Follow this reasoning process STRICTLY:
1. Analyze Context: Compare the new question with the retrieved
examples. Identify similarities in topic, structure, and
complexity.
2. Performance Comparison: For each example, note which model
(LLM or Agent) produced the better answer, considering both content
quality and response time.
3. Pattern Inference: Infer general patterns -- for example, does
the Agent perform better on reasoning-heavy or multi-step questions,
while LLM excels on direct factual queries?
4. Decision Reasoning: Decide which model should handle the new
question and explain your reasoning.
5. Final Decision: Output EXACTLY one of the following on the last
line:
FINAL ANSWER: YES # use Agent
FINAL ANSWER: NO # use LLM
Your final answer must be either "YES" or "NO".
```

Figure 7: Comparison between regular CoT and rubric-guided CoT prompts.

A.4 PROMPTS FOR ABLATION STUDY

Prompt Routing

You are an intelligent routing system that determines which model should answer a question.
Model A:

- Strengths: A fast, stable, general-purpose QA model that excels at natural language understanding, straightforward reasoning, and well-formatted outputs; ideal for simple to medium tasks.

Model B:

- Strengths: A self-evolving reasoning agent capable of complex multi-step planning, self-consistency checking, and structured problem solving; slower but stronger in deep reasoning tasks.

Choose the most suitable model based only on these capability profiles and the question below.
Question: {original_question}
Please answer only: **YES** (use Model B) or **NO** (use Model A).

Routing Prompt

You are an intelligent routing system that determines which model should answer a question.
Original question: {original_question}
Retrieved similar question examples:
{chr(10).join(reference_examples) if reference_examples else "No similar questions found"}
Based on the similar questions and their historical performance, decide which model should answer the original question.
Please answer only: **YES** (use Model B) or **NO** (use Model A).