

EvoRoute: Experience-Driven Self-Routing LLM Agent Systems

Guibin Zhang¹, Haiyang Yu², Kaiming Yang¹, Bingli Wu²,
 Fei Huang², Yongbin Li^{2†}, Shuicheng Yan^{1†}

¹National University of Singapore, ²Tongyi Lab, [†]Corresponding Authors

Main Contact: guibinz@outlook.com

Abstract

Complex agentic AI systems, powered by a coordinated ensemble of Large Language Models (LLMs), tool and memory modules, have demonstrated remarkable capabilities on intricate, multi-turn tasks. However, this success is shadowed by prohibitive economic costs and severe latency, exposing a critical, yet under-explored, trade-off. We formalize this challenge as the **Agent System Trilemma**: the inherent tension among achieving state-of-the-art performance, minimizing monetary cost, and ensuring rapid task completion. To dismantle this trilemma, we introduce **EvoRoute**, a self-evolving model routing paradigm that transcends static, pre-defined model assignments. Leveraging an ever-expanding knowledge base of prior experience, **EvoRoute** dynamically selects Pareto-optimal LLM backbones at each step, balancing accuracy, efficiency, and resource use, while continually refining its own selection policy through environment feedback. Experiments on challenging agentic benchmarks such as GAIA and BrowseComp+ demonstrate that **EvoRoute**, when integrated into off-the-shelf agentic systems, not only sustains or enhances system performance but also reduces execution cost by up to 80% and latency by over 70%.

1 Introduction

As Large Language Model (LLM)-powered agents continue to demonstrate increasingly advanced cognitive capabilities, spanning perception (Driess et al., 2023; Zheng et al., 2023; Wei et al., 2024), planning (Zhu et al., 2024; Erdogan et al., 2025), reasoning (Putta et al., 2024; Masterman et al., 2024), and action (Li et al., 2024; Yang et al., 2024), LLM-driven agentic AI systems have achieved remarkable performance across a range of highly complex, multi-turn tasks, including machine learning engineering (Chan et al., 2025), multi-hop information searching (Mialon et al., 2023), report

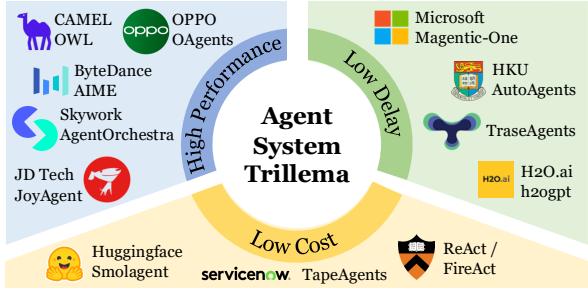


Figure 1: The agent system trilemma. Existing (deep research) agentic systems excel in certain aspects, yet none of which can fulfill the three characteristics spontaneously.

generation (Chen et al., 2025a), and GitHub issue repair (Jimenez et al., 2024).

Moving beyond single-agent frameworks, these agentic systems are distinguished by the coordinated operation of multiple heterogeneous models (potentially developed by different institutions or specialized for distinct domains) alongside sophisticated tool invocation workflows and memory modules (Chen et al., 2025b; Chaudhry et al., 2025). Prominent recent instances include AgentOrchestra (Zhang et al., 2025c) from Skywork AI, which achieved a state-of-the-art (SOTA) open-source score of 70%+ on GAIA (Mialon et al., 2023), and ML-master (Liu et al., 2025), which obtained 29.3% on MLE-Bench (Chan et al., 2025), both of which involve the utilization of cutting-edge models (e.g., OpenAI GPT-4.1 (OpenAI), Gemini-2.5-Pro (Comanici et al., 2025)).

Despite their impressive performance, several underexplored evaluation dimensions raise concerns about the practical deployment of such systems. Empirical studies have revealed that certain general-purpose agentic systems, such as OWL-Workforce (Hu et al., 2025), which achieved open-source SOTA on GAIA in June 2025, incur substantial execution costs, averaging up to \$3 per task. In some cases, even relatively simple queries may entail up to 40 minutes of execution, despite ultimately arriving at the correct answer. Similarly,

R&D Agent (Yang et al., 2025) failed to complete the 75 data science tasks specified by MLE-Bench within a 24-hour window. In summary, the ♣ **efficiency** (*i.e.*, task completion speed) and the ♠ **cost** (*i.e.*, cumulative expenditure on LLM invocations and tool usage) of compound agentic systems often fall short of their promising ♦ **performance** metrics, as shown in Figure 1.

To explicitly formalize this challenge, we draw inspiration from the classical “impossible trinity” theory in economics (Aizenman et al., 2013) and blockchain (Koutsouapis, 2021), where it is deemed infeasible for a system to simultaneously achieve all three desirable properties (*e.g.*, scalability, security, and decentralization in the blockchain context). Analogously, we propose the **Agent System Trilemma** to characterize the inherent trade-offs in complex agentic systems. Specifically, when functioning as general-purpose AI assistants, such systems often face an intrinsic tension among three core objectives: **performance** (*i.e.*, task success and accuracy), **efficiency** (*i.e.*, time or steps required to complete tasks), and **cost** (*i.e.*, computational and monetary resources consumed).

Is it truly impossible to devise an approach capable of overcoming this trilemma and achieving a *tri-optimal* agentic system? We posit that the answer is affirmative. A particularly promising direction lies in model routing (Hu et al., 2024a), *i.e.*, intelligently selecting the most suitable LLM agent from a pool of candidate models given a task query. However, existing model routing paradigms remain insufficient to break the trilemma. The first, ➤ **single model routing**, *i.e.*, assigning one model to handle the entire task (Feng et al., 2024; Chen et al., 2024), works for atomic tasks but struggles with complex agentic workflows, where sub-tasks like web browsing, coding, and summarization often require distinct model capabilities (Chen et al., 2025b; Frick et al., 2025). The second paradigm, ➤ **multi-agent routing**, formally defined in (Yue et al., 2025; Chen et al., 2025b), selects different models for different agent roles within a multi-agent system. Although this approach aligns more closely with the structure of agentic systems, it has so far been limited to relatively simple, single-step reasoning scenarios (*e.g.*, math reasoning and code generation), and relies heavily on large-scale trajectories to model agent behavior, making it ill-suited for dynamic, multi-turn agentic scenarios.

To confront this trilemma, we introduce **EvoRoute**, a tri-optimal, self-evolving model rout-

ing paradigm for general-purpose agentic systems. Specifically, rather than committing to a single model or a fixed multi-agent configuration, **EvoRoute** operates at the granularity of individual sub-tasks within a complex workflow. Before executing each step, it dynamically selects the most judicious LLM by: ❶ **retrieval**, performing a multi-faceted retrieval to identify historically analogous sub-task executions from an evolving knowledge base; ❷ **filtration**, distilling a Pareto-optimal set of candidate models, *i.e.*, those that are not dominated across the axes of cost, efficiency, and performance; and ❸ **selection**, leveraging a lightweight decision model to make the final selection based on this rich, context-aware statistical evidence.

The “self-evolving” nature of **EvoRoute** is realized through a dual-phase operational design: during the Optimization Phase, the system engages in a tree-based exploration, sampling multiple trajectories for a given task to proactively populate and diversify its knowledge base on model behaviors; conversely, during the Inference Phase, it leverages this accumulated wisdom to pursue a single, optimized execution path, ensuring rapid and cost-effective task completion. This adaptive, experience-driven approach allows **EvoRoute** to continuously refine its routing strategy, evolving toward breaking the constraints of the Agent System Trilemma.

Our contributions can be summarized as follows:

❶ **Problem Formulation.** We formalize the critical bottleneck in current agentic systems as the *Agent System Trilemma*, an intrinsic trade-off among performance, cost, and efficiency, and introduce a viable approach to mitigate the issue empirically.

❷ **Technical Solution.** We propose **EvoRoute**, a novel self-evolving routing paradigm that dismantles this trilemma through fine-grained model selection, which combines multi-faceted retrieval with Pareto-optimality selection to make resource-aware model routing.

❸ **Empirical Validation.** We conduct extensive experiments on five challenging benchmarks, including GAIA and BrowseComp+, and demonstrate that **EvoRoute** can outperform vanilla agent systems by up to 10.3% while incurring only $\sim 20\%$ of the cost and achieving nearly $3\times$ faster execution.

2 Related Work

Agentic AI Systems Contemporary multi-agent systems can be broadly categorized by their level of automation into three classes: ■ **Handcrafted**, where the entire system configuration (e.g., LLM backbone, prompting strategies, and communication protocols) is manually specified represented by AutoGen (Wu et al., 2023), AutoGPT (Richards and et al., 2023), Camel (Li et al., 2023), and ChatDev (Qian et al., 2023); ■ **Partially Automated**, which automate specific system components: for example, AutoAgent (Chen et al., 2023), LLMSelector (Chen et al., 2025b), and MasRouter (Yue et al., 2025) automate agent role assignment; DsPy (Khattab et al., 2023) and TextGrad (Yuksekgonul et al., 2024) optimize prompt design; GPTSwarm (Zhuge et al., 2024) and G-Designer (Zhang et al., 2024a) adaptively construct inter-agent topologies; ■ **Fully Automated**, where all modules within the system are autonomously designed and evolved (Hu et al., 2024b; Zhang et al., 2024b, 2025b; Wu et al., 2025; Nie et al., 2025; Gao et al., 2025; Zhang et al., 2025a).

LLM & Agent Routing Leveraging multiple models via intelligent routing has emerged as a central paradigm in modern AI and ML, aiming to exploit complementary model capabilities to enhance task performance while potentially optimizing computational costs (Srivatsa et al., 2024). Router-based approaches, which constitute the primary focus of this work, learn to assign each query to the most appropriate model (Hu et al., 2024a). Classical approaches can be categorized as: (I) neural network-based routers trained on performance or cost signals, including LLM-Blender, which aggregates outputs from top- k models selected via pairwise comparisons (Jiang et al., 2023), ZOOTER, which enhances router training with reward-guided, tag-based label augmentation (Lu et al., 2023), and others (Ong et al., 2024; Feng et al., 2024; Zhang et al., 2025d); (II) cluster-based methods, including UniRoute (Jitkrittum et al., 2025), BEST-Route (Ding et al., 2025), Avengers (Zhang et al., 2025e) and also the baselines introduced in Router-Bench (Hu et al., 2024a). Despite their success, these methods predominantly focus on single-turn responses and are evaluated on relatively simple benchmarks (e.g., Chatbot Arena (Chiang et al., 2024), MATH (Hendrycks et al., 2021)), and they have yet to achieve fine-grained routing in more complex scenarios such as deep research tasks.

3 Preliminary

In this section, we provide a general definition of LLM-based agentic AI systems and their operational workflow, and then formally define the objective of the model routing within this context.

Notations. We consider a complex agentic AI system, \mathcal{M} , designed to resolve a user-issued query \mathcal{Q} through a multi-step workflow, which is typically composed of a set of specialized agents or roles, $\mathcal{I} = \{1, 2, \dots, N\}$, which operate sequentially under a turn-based protocol where a scheduler determines the active agent at each time step. The workflow can involve a series of actions ranging from task decomposition to the execution of specific sub-tasks, formally defined as:

$$\mathcal{M} = \langle \mathcal{I}, \mathcal{L}, \phi, \mathcal{S}, \mathcal{T}, \mathcal{A}, \Psi, \mu, \mathcal{Q} \rangle, \quad (1)$$

where $\mathcal{I} = \{1, 2, \dots, N\}$ denotes the set of agent roles (e.g., web-browser, coder), and \mathcal{L} represents the pool of available LLM backbones. Each agent $i \in \mathcal{I}$ is statically assigned an LLM via a mapping $\phi : \mathcal{I} \rightarrow \mathcal{L}$. The system state is maintained in \mathcal{S} , typically implemented as a shared memory or scratchpad. The agentic workflow may invoke a set of external tools \mathcal{T} , such as code interpreters (Dong et al., 2023) or web search APIs (Jimenez et al., 2024). The full action space \mathcal{A} includes both natural language actions and tool invocations, formally $\mathcal{A} = \mathcal{A}_{\text{lang}} \cup \{\text{use_tool}(T, \text{args}) \mid T \in \mathcal{T}\}$. The transition dynamics of the system are governed by $\Psi(s_{t+1} \mid s_t, a_t)$, while the scheduler $\mu(t) \in \mathcal{I}$ selects the active agent at each time step t .

At each step t , the active agent $i = \mu(t)$ generates an action a_t . This action is produced by its policy π_i , which is instantiated by its designated LLM, $\phi(i) \in \mathcal{L}$. The policy takes into account the current state s_t , the overall query \mathcal{Q} , and the relevant interaction history \mathcal{H}_t :

$$a_t \sim \pi_i(s_t, \mathcal{H}_t, \mathcal{Q}), \quad i = \mu(t), \quad \pi_i \leftarrow \phi(i). \quad (2)$$

The full execution trajectory of the system is thus a sequence of states and actions:

$$\tau = (s_0, a_0, s_1, a_1, \dots, s_T), \quad (3)$$

where T is the terminal step. The final answer to query \mathcal{Q} is synthesized from the information aggregated throughout this trajectory τ .

A critical and often implicit characteristic of these established agentic systems is that the agent-to-model mapping, ϕ , is *static and human-predefined*. For instance, a Web-Browser agent

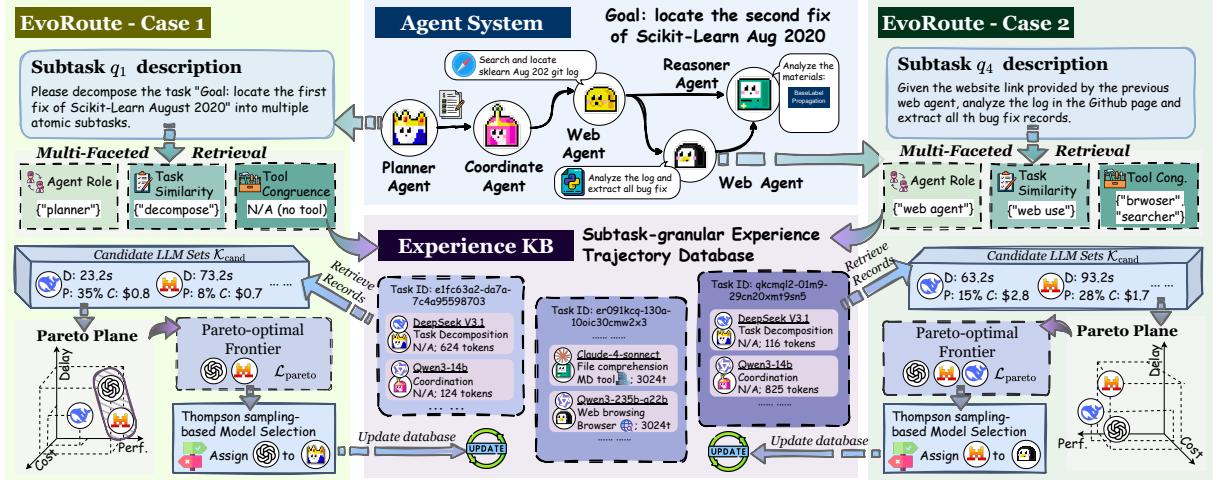


Figure 2: The overview of our proposed **EvoRoute**.

is hard-coded to use GPT-4o in OWL-Workforce, and o3-mini for a planner agent. Other mainstream frameworks also follow similar patterns, including Microsoft’s Magentic-one (Fourney et al., 2024), Skywork AI’s AgentOrchestra (Zhang et al., 2025c), Bytedance’s AIME (Shi et al., 2025b), and Tencent’s CK-Pro (Fang et al., 2025). This rigid design choice, while straightforward, lacks adaptability and fails to account for the varying difficulty and nature of sub-tasks.

Objective Formulation. This paper is dedicated to transcending the limitations of a static agent-to-model mapping by learning a dynamic routing policy, ρ^* , that optimally navigates the Agent System Trilemma. We formulate this as the following multi-objective optimization problem:

$$\rho^* = \underset{\rho}{\operatorname{argmax}}(\mathbb{E}_{\tau \sim \rho}[\mathbb{P}(\tau)], -\mathbb{E}_{\tau \sim \rho}[\mathbb{C}(\tau)], -\mathbb{E}_{\tau \sim \rho}[\mathbb{D}(\tau)]), \quad (4)$$

where ρ is the dynamic routing policy that selects an LLM $l_t \in \mathcal{L}$ for the active agent at each step t . The expectation $\mathbb{E}_{\tau \sim \rho}$ is taken over the distribution of execution trajectories τ generated under this policy. For each trajectory, we measure three key metrics: ♦ **performance** $\mathbb{P}(\tau)$, the task success score; ♠ **cost** $\mathbb{C}(\tau)$, the cumulative monetary and computational expenditure; and ♣ **efficiency** $\mathbb{D}(\tau)$, the total wall-clock execution time. The pursuit of ρ^* is therefore equivalent to finding a Pareto-optimal solution that maximizes performance while concurrently minimizing cost and delay.

4 Methodology

4.1 Self-Evolving Experience Base

As shown in Figure 2, the cornerstone of **EvoRoute**’s adaptive intelligence lies in its ability to accumulate and leverage past experience. This

is enabled by a self-evolving experience base, denoted as \mathcal{K} , which meticulously logs execution data at the step level. We leverage an exploration strategy to address the cold start issue of \mathcal{K} , with details in Section A. Upon the completion of an agentic workflow for a query \mathcal{Q} , the resulting trajectory $\tau = (s_0, a_0, \dots, s_T)$ is retrospectively dissected. For each step $t \in \{0, \dots, T-1\}$, we extract and persist a detailed record \mathcal{R}_t that encapsulates the full context and outcome of that specific action. Formally, each record \mathcal{R}_t is a structured tuple containing multi-faceted information essential for navigating the trilemma:

$$\mathcal{R}_t = \langle i_t, l_t, q_t, \mathbf{e}_t, T_t, c_t, d_t, \sigma_t, \mathbb{P}(\tau) \rangle, \quad (5)$$

where $i_t = \mu(t) \in \mathcal{I}$ denotes the identifier of the active agent role (e.g., coder or coordinator), while $l_t \in \mathcal{L}$ specifies the particular LLM backbone selected for that role at step t . The element q_t represents the natural language sub-task instruction provided to the agent, and its embedding $\mathbf{e}_t = \text{Embed}(q_t)$ is used for semantic similarity retrieval. $T_t \subseteq \mathcal{T}$ is the subset of tools invoked in the action a_t , with $T_t = \emptyset$ if no tools are used. The terms c_t and d_t denote the monetary cost and wall-clock duration incurred during this step, respectively. $\sigma_t \in \{0, 1\}$ is a binary indicator of execution success (e.g., whether a tool call executed without error), and $\mathbb{P}(\tau)$ is the task-level success rate, linking the contribution of step t to the global task outcome.

After each full task execution, the knowledge base \mathcal{K} is dynamically updated by appending all newly generated records:

$$\mathcal{K} \leftarrow \mathcal{K} \cup \{\mathcal{R}_t\}_{t=0}^{T-1}. \quad (6)$$

This granular, step-wise logging ensures that \mathcal{K} evolves into a rich repository capturing the nuanced interplay between sub-task characteristics, model choices, and their resultant impact on performance, cost, and efficiency. This accumulated empirical knowledge forms the foundation for the subsequent retrieval and selection stages.

4.2 Multi-Faceted Retrieval

With the knowledge base \mathcal{K} established, the first active stage of **EvoRoute** is to retrieve a diverse set of historically analogous records when a new sub-task arises. Given a new sub-task at step t' , characterized by the active agent $i_{t'} = \mu(t')$ and its instruction $q_{t'}$, our goal is to gather a comprehensive candidate set $\mathcal{K}_{\text{cand}} \subseteq \mathcal{K}$. Instead of enforcing a strict intersection of criteria, our approach aggregates records that match on *at least one* of three key facets: agent role, semantic similarity, or tool-use profile. This disjunctive strategy ensures a broad and varied pool of evidence for the subsequent decision-making process. Formally, the final candidate set $\mathcal{K}_{\text{cand}}$ is constructed as the union of three independently retrieved subsets:

$$\mathcal{K}_{\text{cand}} = \mathcal{K}_{\text{agent}} \cup \mathcal{K}_{\text{sem}} \cup \mathcal{K}_{\text{tool}}, \quad (7)$$

where each subset is retrieved based on a distinct relevance facet:

➤ **Agent Role Matching** identifies all historical steps performed by the same agent role, capturing functionally equivalent precedents.

$$\mathcal{K}_{\text{agent}} = \{\mathcal{R}_t \in \mathcal{K} \mid i_t = i_{t'}\} \quad (8)$$

➤ **Semantic Similarity Retrieval** retrieves records of sub-tasks that are semantically close to the current one $q_{t'}$. Using a pre-trained sentence encoder $\text{Embed}(\cdot)$ and a similarity metric (*e.g.*, cosine similarity), we select all records whose instruction embeddings \mathbf{e}_t surpass a threshold θ_{sim} :

$$\mathcal{K}_{\text{sem}} = \{\mathcal{R}_t \in \mathcal{K} \mid \text{sim}(\text{Embed}(q_{t'}), \mathbf{e}_t) \geq \theta_{\text{sim}}\}, \quad (9)$$

where $\text{Embed}(\cdot)$ is implemented via MiniLM (Wang et al., 2020) and $\text{sim}(\cdot, \cdot)$ adopts cosine similarity.

➤ **Tool Congruence Retrieval** gathers records based on operational similarity, targeting sub-tasks that likely require similar tool interactions. This facet is specifically designed for instructions $q_{t'}$ where tool usage is anticipated. We first employ a lightweight prediction function, $\text{PredictTools}(q_{t'})$ (whose implementation is detailed in Section B), to

analyze the instruction and infer a set of probable tools, denoted as T'_{pred} . A historical record \mathcal{R}_t is then retrieved if its set of invoked tools, T_t , has a non-empty intersection with this predicted set:

$$\mathcal{K}_{\text{tool}} = \{\mathcal{R}_t \in \mathcal{K} \mid T_t \cap \text{PredictTools}(q_{t'}) \neq \emptyset\}. \quad (10)$$

By taking the union of these three sets, **EvoRoute** assembles a rich and multi-dimensional collection of precedents, $\mathcal{K}_{\text{cand}}$. This set forms the empirical foundation for the subsequent filtration stage, where we will derive model-specific statistics and identify the Pareto-optimal frontier.

4.3 Pareto-Optimal Filtration and Selection

Having retrieved a candidate set of historical records $\mathcal{K}_{\text{cand}}$, the final stage of **EvoRoute** distills this empirical evidence into a decisive action. The process begins by identifying the unique LLMs, $\mathcal{L}_{\text{cand}} = \{l_t \mid \mathcal{R}_t \in \mathcal{K}_{\text{cand}}\}$, that have appeared in the retrieved records. For each candidate LLM $l \in \mathcal{L}_{\text{cand}}$, we compute its aggregated trilemma metrics by averaging over the relevant subset of records $\mathcal{K}_{\text{cand}}(l) = \{\mathcal{R}_t \in \mathcal{K}_{\text{cand}} \mid l_t = l\}$. This yields a statistical profile $(\hat{\mathbb{P}}(l), \hat{\mathbb{C}}(l), \hat{\mathbb{D}}(l))$, where $\hat{\mathbb{P}}(l) = \sum_{\mathcal{R}_t \in \mathcal{K}_{\text{cand}}(l)} \mathbb{P}(\tau) \cdot \sigma_t / |\mathcal{K}_{\text{cand}}(l)|$, $\hat{\mathbb{C}}(l) = \sum_{\mathcal{R}_t \in \mathcal{K}_{\text{cand}}(l)} c_t / |\mathcal{K}_{\text{cand}}(l)|$, and $\hat{\mathbb{D}}(l) = \sum_{\mathcal{R}_t \in \mathcal{K}_{\text{cand}}(l)} d_t / |\mathcal{K}_{\text{cand}}(l)|$. Subsequently, we perform Pareto filtration. An LLM l is considered dominated if another LLM l' exists that is superior or equal on all three axes (higher performance, lower cost, lower duration) and strictly superior on at least one. By retaining only the non-dominated models, we form the Pareto-optimal set, $\mathcal{L}_{\text{pareto}}$.

A purely greedy selection from $\mathcal{L}_{\text{pareto}}$ would stifle exploration. To address this, we employ *Thompson sampling* (Russo et al., 2020). We treat each of the three trilemma metrics as a continuous variable drawn from a Normal distribution and model the uncertainty over its mean and variance using a Normal-Inverse-Gamma (NIG) conjugate prior.

For each model $l \in \mathcal{L}_{\text{pareto}}$, we dynamically construct its posterior distributions based on the retrieved evidence in $\mathcal{K}_{\text{cand}}(l)$. We first compute the sample statistics for each metric $m \in \{\mathbb{P}, \mathbb{C}, \mathbb{D}\}$: the count n_l , the sample mean $\bar{x}_{m,l}$, and the sample variance $s_{m,l}^2$. These statistics are used to parameterize the NIG posteriors, $\text{NIG}(\mu_{m,l}, \nu_{m,l}, \alpha_{m,l}, \beta_{m,l})$, where $\mu_{m,l} = \bar{x}_{m,l}$, $\nu_{m,l} = n_l$, $\alpha_{m,l} = n_l/2$, and $\beta_{m,l} = (n_l - 1)s_{m,l}^2/2$, assuming uninformative priors. At decision time, we draw one sample from each model's

Table 1: Performance comparison on the GAIA and BrowseComp+ benchmarks against both manual and routing-based baselines. For GAIA, results are reported separately for each level, which are defined according to the difficulty of queries by the original benchmark.

Setting	LLM	GAIA (All levels)			Level 1			Level 2			Level 3			BrowseComp+		
		Perf. (%)	Cost (\$)	Delay (h)	Perf. (%)	Cost (\$)	Delay (h)	Perf. (%)	Cost (\$)	Delay (h)	Perf. (%)	Cost (\$)	Delay (h)	Perf. (%)	Cost (\$)	Delay (h)
CK-Pro	Qwen3-14b	10.91	4.16	24.67	16.77	1.06	6.23	10.34	2.58	14.90	0.00	0.52	3.54	6.50	12.50	6.80
	Claude-4	58.28	359.32	45.14	67.92	96.46	6.83	58.14	192.64	25.35	37.50	70.22	12.96	33.50	220.50	20.15
	GPT-4o	42.42	82.86	35.43	59.91	27.56	7.35	39.22	43.86	21.50	15.26	11.44	6.58	24.28	113.46	9.43
	GPT-4.1	48.48	61.84	23.04	56.29	16.96	7.46	49.73	35.26	10.87	26.77	9.62	4.71	31.44	185.01	18.69
	PromptLLM	47.88	128.15	57.70	58.91	31.27	10.88	47.65	80.24	30.93	24.39	16.64	15.89	26.87	133.22	17.27
	Routing GraphRouter	46.67	116.10	53.32	58.13	28.50	10.20	46.23	72.10	28.90	22.94	15.50	14.22	27.90	125.80	16.90
	MasRouter	53.50	120.15	52.95	67.92	27.80	9.80	49.74	77.40	28.15	23.51	14.95	15.00	28.50	128.90	17.10
	Ours EvoRoute	63.19	85.40	38.75	83.02	26.50	6.10	59.30	33.60	18.30	33.33	25.30	14.35	38.72	79.30	10.33
Smolagent	Qwen3-14b	11.52	0.82	8.60	15.35	0.30	3.00	4.73	0.42	1.20	27.39	0.10	4.40	5.80	1.95	4.25
	Manual Claude-4	46.06	76.90	29.10	56.26	21.20	12.80	46.23	41.60	14.50	22.94	14.10	1.80	28.20	85.60	16.50
	GPT-4.1	39.39	18.10	7.90	55.00	5.70	3.20	36.30	8.90	2.60	16.01	3.50	2.10	25.50	58.20	11.20
	PromptLLM	40.00	17.50	12.20	49.80	4.10	6.50	40.92	8.90	3.80	15.04	4.50	1.90	23.10	45.80	10.80
	Routing GraphRouter	38.79	15.70	11.30	47.95	3.70	6.00	39.78	8.00	3.50	15.03	4.00	1.80	24.20	42.50	10.50
	MasRouter	40.61	17.00	12.30	49.82	4.00	6.80	42.07	8.50	3.70	15.04	4.50	1.80	24.80	44.10	10.90
	Ours EvoRoute	56.36	15.60	8.90	73.91	3.70	2.90	53.73	6.20	4.80	27.04	5.70	1.20	32.50	25.50	9.80

full posterior for each metric to generate a stochastic realization of its utility. The model l^* with the highest sampled utility is selected:

For each $l \in \mathcal{L}_{\text{pareto}}$ and each metric $m \in \{\mathbb{P}, \mathbb{C}, \mathbb{D}\}$:

$$\begin{aligned} (\tilde{\mu}_{m,l}, \tilde{\sigma}_{m,l}^2) &\sim \text{NIG}(\mu_{m,l}, \nu_{m,l}, \alpha_{m,l}, \beta_{m,l}) \\ \tilde{x}_{m,l} &\sim \mathcal{N}(\tilde{\mu}_{m,l}, \tilde{\sigma}_{m,l}^2), \\ U'(l) &= w_p \cdot \tilde{x}_{\mathbb{P},l} - w_c \cdot \tilde{x}_{\mathbb{C},l} - w_d \cdot \tilde{x}_{\mathbb{D},l} \\ l^* &= \underset{l \in \mathcal{L}_{\text{pareto}}}{\text{argmax}} (U'(l)), \end{aligned} \quad (11)$$

where the weights (w_p, w_c, w_d) reflect the desired trilemma trade-off.

Crucially, this selection is not the end of the process. Once the agent powered by l^* completes its action, the observed outcome is logged back into the knowledge base \mathcal{K} . This closes the feedback loop, ensuring that every decision and its outcome contribute to the system’s ever-improving wisdom, thereby realizing the self-evolving nature of **EvoRoute**.

5 Experiment

5.1 Experiment Setup

Frameworks. We select three representative agent frameworks, ordered by increasing architectural complexity: ① **ReAct** (Yao et al., 2023), a single-agent architecture that iteratively alternates between “observation–action–reasoning”; ② **Smolagents**¹, a dual-agent framework comprising a manager agent and a tool agent; and ③ **Cognitive**

Kernel-Pro (Fang et al., 2025) by Tencent, which achieved open-source SOTA on GAIA in August 2025 and features a hierarchical multi-agent structure (*e.g.*, main agent, web agent, and file agent.)

Baselines. We compare against two main categories of baselines: **(i) manual setting**, which includes each framework’s original predefined LLM configuration as well as manually specified setups (*e.g.*, uniformly using gpt-4.1 or Gemini-2.5-pro). The choice of manually assigned LLMs may vary across frameworks depending on compatibility requirements. **(ii) model routing**, encompassing SOTA routing methods like PromptLLM (Feng et al., 2024), MasRouter (Yue et al., 2025), and GraphRouter (Feng et al., 2024)².

Method Configurations. We fill our LLM pool \mathcal{L} with models of varying prices and sizes: {Qwen-14b, GPT-4.1, GPT-4o, Claude-4-Sonnet ((Claude-4)), Gemini-2.5-Flash, Gemini-2.5-Pro}. The model prices are listed in Section C.

Parameter Configurations. For our multi-faceted retrieval, we set the semantic similarity threshold θ_{sim} to 0.85 to balance retrieval recall and precision. In the selection stage, we leverage uninformative NIG priors ($\nu_0 = 0, \alpha_0 = 0.5, \beta_0 = 0$) for each metric. The final selection is guided by a utility function with manually configured weights

¹<https://github.com/huggingface/smolagents>

²Our code will be available at <https://github.com/bingreeky/evo-route>.

Table 2: Performance comparison on HotpotQA, DS-1000 and DDXPlus benchmarks.

Setting	Method/Model	DS-1000			HotpotQA			DDXPlus			Avg.		
		Perf.	Cost\$	Delay(h)	Perf.	Cost\$	Delay(h)	Perf.	Cost\$	Delay(h)	Perf.	Cost\$	Delay(h)
Manual	GPT-4o	41.40	54.00	12.32	83.84	391.98	67.29	57.58	135.10	34.31	60.94	581.08	113.92
	GPT-4.1	29.40	49.59	12.01	87.00	213.22	78.77	55.30	89.82	21.95	57.23	352.63	112.73
	Gemini-2.5-pro	54.30	68.33	16.17	87.40	343.43	79.18	81.10	109.85	27.65	74.27	521.61	123.00
	Qwen3-14b	38.20	12.68	10.86	81.40	22.96	35.52	41.26	14.16	31.67	53.62	49.80	78.05
Routing	PromptLLM	52.00	26.66	11.09	85.20	47.02	62.46	60.07	127.05	26.88	65.76	200.73	100.43
	GraphRouter	52.80	24.50	11.50	86.10	64.80	61.80	62.50	119.50	27.50	67.13	208.80	100.80
	MasRouter	53.50	25.10	11.80	88.50	59.50	53.10	73.10	92.38	32.30	71.70	176.98	97.20
Ours	EvoRoute	56.50	23.50	11.20	87.80	49.10	60.50	79.50	65.80	20.53	74.60	138.40	92.23

$(w_p, w_c, w_d) = (1.0, 0.1, 0.05)$, which prioritizes performance while still penalizing cost and delay.

Benchmark and Evaluations. We evaluate our approach on five benchmarks. Two are deep research benchmarks: **GAIA** (Mialon et al., 2023), a widely used general-purpose agentic benchmark that assesses capabilities such as file reading, web browsing, and coding, and is divided into three tiers based on difficulty; and **BrowseComp+** (Chen et al., 2025c), an enhanced version of OpenAI’s BrowseComp (Wei et al., 2025) that corrects erroneous cases and provides more stable evaluations. The remaining three are from StreamBench (Wu et al., 2024): **DS-1000** (Lai et al., 2022) for data science tasks, **HotpotQA** (Yang et al., 2018) for web search, and **DDXPlus** (Tchango et al., 2022) for medical reasoning. The statistics of above datasets are placed in Section D.

5.2 Main Results

Tables 1 and 2 present the comparison of **EvoRoute** against different baselines across three key dimensions: *performance*, *cost*, and *latency*. Figure 3 further illustrates a radar visualization of results across different levels of GAIA. Our main findings are summarized as follows.

Existing Agent Systems Grapple with the Performance-Efficiency-Economy Trilemma. Our empirical analysis reveals that contemporary agentic systems fundamentally struggle to reconcile performance, cost, and latency. Achieving state-of-the-art performance typically incurs exorbitant costs and significant delays. For instance, using the powerful Gemini-2.5-Pro+ReAct achieves a leading 74.27% average performance, but at a prohibitive cost of \$521.61 and a delay of 123 hours, as shown in Table 2. Similarly, Claude-4+CK-Pro reaches a strong 55.15% on GAIA, yet demands an unsustainable \$359.32

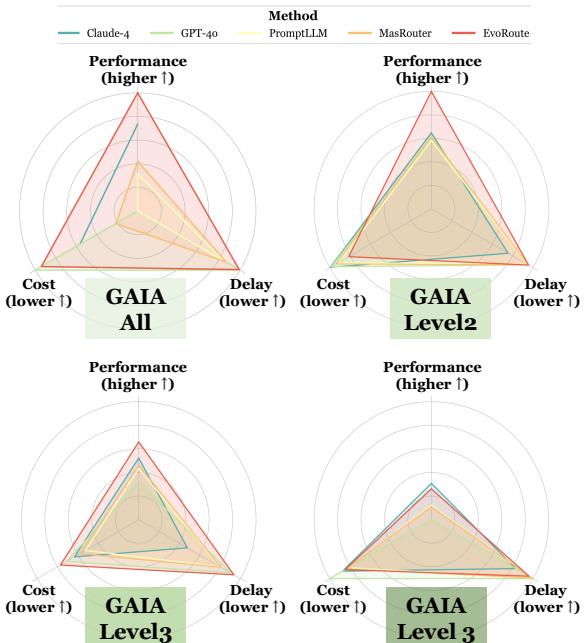


Figure 3: Comparative analysis across three key metrics: performance, cost, and delay, on all subsets of GAIA. All metrics are globally normalized, and values for cost/delay are inverted, such that a larger enclosed area signifies better economy/efficiency.

(Table 1). Conversely, opting for economical models like Qwen3-14b+CK-Pro drastically reduces cost to just \$4.16, but at the expense of a sharp performance drop to 10.91%. While existing SOTA routing methods like MasRouter offer some mitigation (achieving 71.70% average performance for \$176.98), they provide only incremental improvements. These methods still lag behind top-performing models in accuracy and fail to fundamentally break the trade-off.

EvoRoute Effectively Alleviates the Trilemma. In contrast, **EvoRoute** demonstrates a remarkable ability to navigate the trilemma, consistently achieving state-of-the-art performance while substantially reducing cost and latency. Integrated with CK-Pro, it surpasses the strong Claude-4 base-

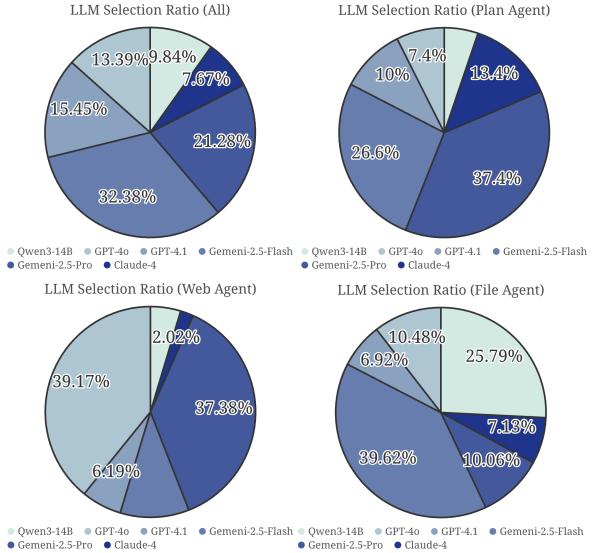


Figure 4: LLM selection distribution of **EvoRoute + CK-Pro** across different agent roles.

line on GAIA (63.18% vs. 58.28%), cuts operational costs by over 76% (\$85.40 vs. \$359.32), and maintains comparable latency. On BrowseComp+, it similarly outperforms Claude-4 (38.72% vs. 33.50%) at under 36% of the cost (\$79.30 vs. \$220.50) and roughly half the latency (10.33h vs. 20.15h). Across all tested configurations (Table 2), **EvoRoute** achieves the highest average performance (74.60%), slightly exceeding Gemini-2.5-Pro (74.27%), while incurring only 26% of the cost (\$138.40 vs. \$521.61) and reducing latency by 25% (92.23h vs. 123.00h). These results firmly establish **EvoRoute** as an effective solution to the agent system trilemma.

5.3 Framework Analysis

Visualization. We analyze **EvoRoute**’s LLM selection across agent roles on CK-Pro+GAIA (Figure 4). The results indicate that model allocation aligns with sub-task complexity: cognitively demanding tasks handled by the Plan Agent favor high-capability models (Gemini-2.5-Pro 37.4%, Claude-4 13.4%), whereas simpler, operational tasks of the File Agent predominantly use cost-efficient models (Qwen3-14B 25.79%), with premium models like Gemini-2.5-Pro and Claude-4 dropping to 10.06% and 7.13%. This shows that **EvoRoute** strategically reserves its most powerful (and expensive) models for critical reasoning, while employing efficient models for routine sub-tasks.

Ablation Study We performed an ablation study on GAIA Level 1 to validate our core components (Table 3). We evaluate four variants: **w/o \mathcal{K} cold start** (starting with an empty knowledge base), **w/o**

Table 3: Ablation study of four variants, each disabling a key component (tested on GAIA Level 1).

Method Variant	Perf. (%)	Cost (\$)	Delay (h)
EvoRoute (Full Model)	83.02	26.50	6.10
w/o \mathcal{K} cold start	69.81 (-13.21)	29.50	6.30
w/o Multi-Faceted Retrieval	71.70 (-11.22)	32.10	6.95
w/o Thompson Sampling	76.50 (-6.52)	29.80	6.40
w/o Pareto Filtration	81.50 (-1.52)	28.20	6.35

Multi-Faceted Retrieval (using only semantic similarity), **w/o Thompson Sampling** (using a greedy policy), and **w/o Pareto Filtration** (omitting model pruning). The absence of a bootstrapped knowledge base is the most detrimental, causing performance to plummet by 13.21% ↓. Disabling the multi-faceted retrieval is nearly as damaging, resulting in an 11.22% performance decrease and incurring the highest cost (\$32.10) and delay (6.95 h) among all variants. Furthermore, replacing Thompson sampling with a greedy policy reduces performance by 6.52%, while omitting Pareto filtration yields a smaller 1.52% drop, confirming that both principled exploration and efficiency-focused pruning are vital to the system’s overall effectiveness.

Architectural Overhead on Simpler Tasks. One may note that, in Section 5.2, ReAct is evaluated on simpler tasks (DS-1000, HotpotQA), while Smolagent and CK-Pro are reserved for GAIA and BrowseComp+. This reflects a practical observation: deploying more complex frameworks on simple tasks can be *counterproductive*. For example, we initially discover that Gemini-2.5-Pro+Smolagent scored only 31.7% on DS-1000, below the 38.2% of the simpler Qwen3-14b+ReAct. Trace analysis suggests that Smolagent’s features (*e.g.*, web search and high-level planning) introduce overhead by overcomplicating tasks solvable with a single LLM I/O. We term this “architectural overfitting,” where structural complexity and specialized tools hinder rather than help. This highlights that, the optimal agent architecture should match the task’s intrinsic complexity, not its maximal sophistication.

6 Conclusion

In this work, we introduced **EvoRoute**, a dynamic model routing framework designed to systematically address the agent system trilemma. Our experiments on challenging benchmarks like GAIA and BrowseComp+ empirically validate this paradigm, delivering substantial reductions in monetary cost (up to 80%) and latency (over 70%) without compromising task success. Ultimately, **EvoRoute** rep-

resents a crucial step towards making powerful agentic AI systems more practical, scalable, and economically viable for real-world deployment.

Limitation & Ethical Concerns

Our evaluation focuses on CK-Pro and Smolagent and does not select other deep-research frameworks such as OWL, Agent-Orchestra, or AIME. Given the proliferation of deep research systems and their broadly similar architectural paradigms—typically consisting of a central coordinator supported by multiple specialized sub-agents—we believe our experiments already capture a representative spectrum of agentic systems from simple to complex. Moreover, the substantial token costs associated with these systems (as shown in Table 1) make exhaustive large-scale benchmarking impractical. Regarding ethical considerations, since our study relies exclusively on standard public benchmarks and widely used open frameworks, we identify no immediate ethical risks associated with this work.

Contributions

G. Zhang was primarily responsible for the method implementation, manuscript preparation, visualization, and experimental analysis. K. Yang conducted a substantial portion of the experimental work and was also one of the core contributors. H. Yu and B. Wu provided extensive guidance and engaged in in-depth discussions throughout the project. F. Huang, Y. Li, and S. Yan offered senior-level supervision and strategic guidance.

References

- Joshua Aizenman, Menzie David Chinn, and Hiro Ito. 2013. The “impossible trinity” hypothesis in an era of global imbalances: Measurement and testing. *Review of international economics*, 21(3):447–458.
- Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Lilian Weng, and Aleksander Madry. 2025. **Mle-bench: Evaluating machine learning agents on machine learning engineering.** *Preprint*, arXiv:2410.07095.
- Gohar Irfan Chaudhry, Esha Choukse, Íñigo Goiri, Rodrigo Fonseca, Adam Belay, and Ricardo Bianchini. 2025. **Towards resource-efficient compound ai systems.** *Preprint*, arXiv:2501.16634.
- Guangyao Chen, Siwei Dong, Yu Shu, Ge Zhang, Jaward Sesay, Börje F. Karlsson, Jie Fu, and Yemin Shi. 2023. **Autoagents: A framework for automatic agent generation.** *CoRR*, abs/2309.17288.
- Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, Chen Sun, Han Hou, Hui Yang, James Pan, Jianan Lou, Jiayi Mao, Jizheng Liu, Jinpeng Li, Kangyi Liu, and 14 others. 2025a. **xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations.** *Preprint*, arXiv:2506.13651.
- Lingjiao Chen, Jared Quincy Davis, Boris Hanin, Peter Bailis, Matei Zaharia, James Zou, and Ion Stoica. 2025b. **Optimizing model selection for compound ai systems.** *Preprint*, arXiv:2502.14815.
- Shuhao Chen, Weisen Jiang, Baijiong Lin, James T Kwok, and Yu Zhang. 2024. **Routerdc: Query-based router by dual contrastive learning for assembling large language models.** *arXiv preprint arXiv:2409.19886*.
- Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Sahel Sharifmoghaddam, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Nandan Thakur, Cristina Zhang, Luyu Gao, Wenhui Chen, and Jimmy Lin. 2025c. **Browsocomp-plus: A more fair and transparent evaluation benchmark of deep-research agent.** *Preprint*, arXiv:2508.06600.
- Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. 2024. **Chatbot arena: An open platform for evaluating llms by human preference.** *Preprint*, arXiv:2403.04132.
- Gheorghe Comanici, Eric Bieber, Mike Schaeckermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. **Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities.** *arXiv preprint arXiv:2507.06261*.
- Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian Del Carmen Hipolito Garcia, Menglin Xia, Laks V. S. Lakshmanan, Qingyun Wu, and Victor Rühle. 2025. **BEST-route: Adaptive LLM routing with test-time optimal compute.** In *Forty-second International Conference on Machine Learning*.
- Yihong Dong, Xue Jiang, Zhi Jin, and Ge Li. 2023. **Self-collaboration Code Generation via ChatGPT.** *arXiv e-prints*, arXiv:2304.07590.
- Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, and 1 others. 2023. **Palm-e: An embodied multi-modal language model.**
- Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. **Plan-and-act:**

- Improving planning of agents for long-horizon tasks. *arXiv preprint arXiv:2503.09572*.
- Tianqing Fang, Zhisong Zhang, Xiaoyang Wang, Rui Wang, Can Qin, Yuxuan Wan, Jun-Yu Ma, Ce Zhang, Jiaqi Chen, Xiyun Li, Hongming Zhang, Haitao Mi, and Dong Yu. 2025. **Cognitive kernel-pro: A framework for deep research agents and agent foundation models training.** *Preprint*, arXiv:2508.00414.
- Tao Feng, Yanzhen Shen, and Jiaxuan You. 2024. **Graphrouter: A graph-based router for llm selections.** *Preprint*, arXiv:2410.03834.
- Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Erkang Zhu, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, Peter Chang, Ricky Loynd, Robert West, Victor Dibia, Ahmed Awadallah, Ece Kamar, Rafah Hosn, and Saleema Amershi. 2024. **Magnetic-one: A generalist multi-agent system for solving complex tasks.** *Preprint*, arXiv:2411.04468.
- Evan Frick, Connor Chen, Joseph Tennyson, Tianle Li, Wei-Lin Chiang, Anastasios N. Angelopoulos, and Ion Stoica. 2025. **Prompt-to-leaderboard.** *Preprint*, arXiv:2502.14855.
- Hongcheng Gao, Yue Liu, Yufei He, Longxu Dou, Chao Du, Zhijie Deng, Bryan Hooi, Min Lin, and Tianyu Pang. 2025. **Flowreasoner: Reinforcing query-level meta-agents.** *Preprint*, arXiv:2504.15257.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.
- Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Bowei Xia, Tao Sun, Ziyu Ye, Zhaoxuan Jin, Yingru Li, Qiguang Chen, Zeyu Zhang, Yifeng Wang, Qianshuo Ye, Bernard Ghanem, Ping Luo, and Guohao Li. 2025. **Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation.** *Preprint*, arXiv:2505.23885.
- Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. 2024a. **Router-bench: A benchmark for multi-lm routing system.** *Preprint*, arXiv:2403.12031.
- Shengran Hu, Cong Lu, and Jeff Clune. 2024b. Automated design of agentic systems. *arXiv preprint arXiv:2408.08435*.
- Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. **LLM-blender: Ensembling large language models with pairwise ranking and generative fusion.** In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada. Association for Computational Linguistics.
- Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. **Swe-bench: Can language models resolve real-world github issues?** *Preprint*, arXiv:2310.06770.
- Wittawat Jitkrittum, Harikrishna Narasimhan, Ankit Singh Rawat, Jeevesh Juneja, Congchao Wang, Zifeng Wang, Alec Go, Chen-Yu Lee, Pradeep Shenoy, Rina Panigrahy, Aditya Krishna Menon, and Sanjiv Kumar. 2025. **Universal model routing for efficient llm inference.** *Preprint*, arXiv:2502.08773.
- Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T Joshi, Hanna Moazam, and 1 others. 2023. **Dspy: Compiling declarative language model calls into self-improving pipelines.** *arXiv preprint arXiv:2310.03714*.
- Dimitrios Koutsoupakis. 2021. Are stable cryptocurrencies exempted from impossible trinity? *Journal of Economic Studies*, 48(8):1480–1496.
- Yuhang Lai, Chengxi Li, Yiming Wang, Tianyi Zhang, Ruiqi Zhong, Luke Zettlemoyer, Scott Wen tau Yih, Daniel Fried, Sida Wang, and Tao Yu. 2022. **Ds-1000: A natural and reliable benchmark for data science code generation.** *Preprint*, arXiv:2211.11501.
- Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbulin, and Bernard Ghanem. 2023. **CAMEL: communicative agents for "mind" exploration of large language model society.** In *NeurIPS*.
- Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Erran Li Li, Ruohan Zhang, and 1 others. 2024. **Embodied agent interface: Benchmarking llms for embodied decision making.** *Advances in Neural Information Processing Systems*, 37:100428–100534.
- Zexi Liu, Yuzhu Cai, Xinyu Zhu, Yujie Zheng, Runkun Chen, Ying Wen, Yanfeng Wang, Weinan E, and Siheng Chen. 2025. **Ml-master: Towards ai-for-ai via integration of exploration and reasoning.** *Preprint*, arXiv:2506.16499.
- Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. **Routing to the expert: Efficient reward-guided ensemble of large language models.** *Preprint*, arXiv:2311.08692.
- Tula Masterman, Sandi Besen, Mason Sawtell, and Alex Chao. 2024. The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey. *arXiv preprint arXiv:2404.11584*.
- Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. **Gaia: a benchmark for general ai assistants.** In *The Twelfth International Conference on Learning Representations*.

- Fan Nie, Lan Feng, Haotian Ye, Weixin Liang, Pan Lu, Huaxiu Yao, Alexandre Alahi, and James Zou. 2025. **Weak-for-strong: Training weak meta-agent to harness strong executors.** *Preprint*, arXiv:2504.04785.
- Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M Waleed Kadous, and Ion Stoica. 2024. **Routellm: Learning to route llms with preference data.** *Preprint*, arXiv:2406.18665.
- OpenAI. [\[link\]](#).
- Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. 2024. Agent q: Advanced reasoning and learning for autonomous ai agents. *arXiv preprint arXiv:2408.07199*.
- Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. 2023. Communicative agents for software development. 25 pages, 9 figures, 2 tables.
- Toran Bruce Richards and et al. 2023. Auto-gpt: An autonomous gpt-4 experiment. <https://github.com/Significant-Gravitas/Auto-GPT>.
- Daniel Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. 2020. **A tutorial on thompson sampling.** *Preprint*, arXiv:1707.02038.
- Dingfeng Shi, Jingyi Cao, Qianben Chen, Weichen Sun, Weizhen Li, Hongxuan Lu, Fangchen Dong, Tianrui Qin, King Zhu, Minghao Liu, Jian Yang, Ge Zhang, Jiaheng Liu, Changwang Zhang, Jun Wang, Yuchen Eleanor Jiang, and Wangchunshu Zhou. 2025a. **Taskcraft: Automated generation of agentic tasks.** *Preprint*, arXiv:2506.10055.
- Yexuan Shi, Mingyu Wang, Yunxiang Cao, Hongjie Lai, Junjian Lan, Xin Han, Yu Wang, Jie Geng, Zhenan Li, Zihao Xia, Xiang Chen, Chen Li, Jian Xu, Wenbo Duan, and Yuanshuo Zhu. 2025b. **Aime: Towards fully-autonomous multi-agent framework.** *Preprint*, arXiv:2507.11988.
- KV Aditya Srivatsa, Kaushal Kumar Maurya, and Ekaterina Kochmar. 2024. **Harnessing the power of multiple minds: Lessons learned from llm routing.** *Preprint*, arXiv:2405.00467.
- Arsene Fansi Tchango, Rishab Goel, Zhi Wen, Julien Martel, and Joumana Ghosn. 2022. **Ddxplus: A new dataset for automatic medical diagnosis.** *Preprint*, arXiv:2205.09148.
- Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.
- Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. **Browsecmp: A simple yet challenging benchmark for browsing agents.** *Preprint*, arXiv:2504.12516.
- Yuxi Wei, Zi Wang, Yifan Lu, Chenxin Xu, Changxing Liu, Hao Zhao, Siheng Chen, and Yanfeng Wang. 2024. Editable scene simulation for autonomous driving via collaborative llm-agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15077–15087.
- Cheng-Kuang Wu, Zhi Rui Tam, Chieh-Yen Lin, Yun-Nung Chen, and Hung yi Lee. 2024. **Streambench: Towards benchmarking continuous improvement of language agents.** *Preprint*, arXiv:2406.08747.
- Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation framework.
- Shirley Wu, Parth Sarthi, Shiyu Zhao, Aaron Lee, Herumb Shandilya, Adrian Mladenic Grobelnik, Nurendra Choudhary, Eddie Huang, Karthik Subbian, Linjun Zhang, and 1 others. 2025. Optimas: Optimizing compound ai systems with globally aligned local rewards. *arXiv preprint arXiv:2507.03041*.
- Xu Yang, Xiao Yang, Shikai Fang, Bowen Xian, Yuante Li, Jian Wang, Minrui Xu, Haoran Pan, Xinpeng Hong, Weiqing Liu, and 1 others. 2025. R&d-agent: Automating data-driven ai solution building through llm-powered automated research, development, and evolution. *arXiv preprint arXiv:2505.14738*.
- Yijun Yang, Tianyi Zhou, Kanxue Li, Dapeng Tao, Lusong Li, Li Shen, Xiaodong He, Jing Jiang, and Yuhui Shi. 2024. Embodied multi-modal agent trained by an llm from a parallel textworld. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26275–26285.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.
- Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. 2025. Masrouter: Learning to route llms for multi-agent systems. *arXiv preprint arXiv:2502.11133*.
- Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. 2024. **Textgrad: Automatic "differentiation" via text.** *Preprint*, arXiv:2406.07496.

Guibin Zhang, Kaijie Chen, Guancheng Wan, Heng Chang, Hong Cheng, Kun Wang, Shuyue Hu, and Lei Bai. 2025a. Evoflow: Evolving diverse agentic workflows on the fly. *arXiv preprint arXiv:2502.07373*.

Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. 2025b. Multi-agent architecture search via agentic supernet. *arXiv preprint arXiv:2502.04180*.

Guibin Zhang, Yanwei Yue, Xiangguo Sun, Guancheng Wan, Miao Yu, Junfeng Fang, Kun Wang, Tianlong Chen, and Dawei Cheng. 2024a. G-designer: Architecting multi-agent communication topologies via graph neural networks. *arXiv preprint arXiv:2410.11782*.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. 2024b. **AFlow: Automating Agentic Workflow Generation.** *arXiv preprint*. ArXiv:2410.10762.

Wentao Zhang, Ce Cui, Yilei Zhao, Rui Hu, Yang Liu, Yahui Zhou, and Bo An. 2025c. **Agentorchestra: A hierarchical multi-agent framework for general-purpose task solving.** *Preprint*, arXiv:2506.12508.

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

Yiqun Zhang, Hao Li, Chenxu Wang, Linyao Chen, Qiaosheng Zhang, Peng Ye, Shi Feng, Daling Wang, Zhen Wang, Xinrun Wang, Jia Xu, Lei Bai, Wanli Ouyang, and Shuyue Hu. 2025e. **The avengers: A simple recipe for uniting smaller language models to challenge proprietary giants.** *Preprint*, arXiv:2505.19797.

Sipeng Zheng, Jiazheng Liu, Yicheng Feng, and Zongqing Lu. 2023. Steve-eye: Equipping llm-based embodied agents with visual perception in open worlds. *arXiv preprint arXiv:2310.13255*.

Yuqi Zhu, Shuofei Qiao, Yixin Ou, Shumin Deng, Shiwei Lyu, Yue Shen, Lei Liang, Jinjie Gu, Hua-jun Chen, and Ningyu Zhang. 2024. Knowagent: Knowledge-augmented planning for llm-based agents. *arXiv preprint arXiv:2403.03101*.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbulin, and Jürgen Schmidhuber. 2024. Gptswarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*.

A Cold Start Issue

A fundamental challenge for any experience-based system is the “cold start” issue: the experience base, \mathcal{K} , is initially empty (*tabula rasa*), rendering the retrieval and routing mechanisms ineffective as

they have no prior data upon which to base their decisions. To address this, we implement a dedicated exploration strategy designed to populate \mathcal{K} with a diverse and informative set of initial experiences before the system is deployed for operational use.

We leverage a curated set of 50 agentic tasks from the TaskCraft dataset (Shi et al., 2025a). For each of these 50 tasks, we execute both the Smolagent and CK-Pro frameworks from start to finish. We employ a stochastic exploration policy to maximize the diversity of the collected data. Specifically, at each step t of a task’s execution, the LLM backbone l_t for the active agent $i_t = \mu(t)$ is selected via uniform random sampling from the entire pool of available models \mathcal{L} . Formally, the selection is made as follows:

$$l_t \sim \mathcal{U}(\mathcal{L}), \quad (12)$$

where $\mathcal{U}(\mathcal{L})$ denotes the uniform distribution over the discrete set of LLM backbones.

Upon the completion of each of the 50 tasks, the full execution trajectory τ is processed as described in our main methodology. The complete set of step-wise records $\{\mathcal{R}_t\}_{t=0}^{T-1}$ is extracted and used to populate the initially empty experience base \mathcal{K} .

We further emphasize that the cold-start process is not costly: it required only \$28.8 and approximately 3.6 hours to complete, yielding around 480 step-level records that provide a sufficiently informative prior. Overall, this initialization overhead remains modest.

B Tool Prediction Function

The tool prediction function $\text{PredictTools}(q_t)$ employs a two-stage hybrid strategy to balance predictive accuracy with minimal computational overhead. Initially, the function performs a near-instantaneous heuristic check, using a predefined dictionary to map explicit trigger keywords (e.g., “search” for `web_search`; “run”, “plot” for `code_interpreter`) to their corresponding tools. This handles the majority of clear-cut cases with zero latency or API cost. If, and only if, this initial heuristic fails to find a match, the function escalates to a more powerful fallback: a single API call to a cheap and effective LLM (practically, Qwen3-14b). This model is prompted in a zero-shot manner to analyze the instruction’s semantics and identify the necessary tools from the available set.

Table 4: The pool of LLM backbones (\mathcal{L}) used in our experiments, along with their respective pricing per one million tokens. The models were selected to cover a diverse range of capabilities and operational costs.

Model	Input Price (\$/M)	Output Price (\$/M)
Gemini-2.5-Flash	\$0.3	\$2.5
Qwen3-14B	\$0.05	\$0.22
Claude-4	\$3.00	\$15.00
GPT-4o	\$2.50	\$10.00
Gemini-2.5-Pro	\$1.25	\$10.00
GPT-4.1	\$2.00	\$8.00

C Model Price

D Dataset Details

The GAIA benchmark (Mialon et al., 2023) offers a broad evaluation suite for general-purpose AI assistants, comprising 165 tasks systematically organized into three difficulty tiers: 53 basic tasks (Level 1), 86 intermediate tasks (Level 2), and 26 advanced tasks (Level 3). The BrowseComp+ benchmark includes 830 evaluation instances, while DS-1000 consists of 1,000 tasks. HotpotQA contains 7,405 queries. For DDXPlus, originally comprising approximately 130K samples, we randomly subsample 1,000 instances for evaluation.

E Use of AI Assistants

We employed AI-based tools, including large language models, to support various stages of manuscript preparation. These tools were used for language polishing, improving clarity and readability, formatting references, and generating visualizations.