RAAS: Relative Architecture Adaptive Search for Agentic Supernet Optimization

Jiayi Yang*, **Mengting Li***, **Guancheng Wan***,

University of California, Los Angeles

Abstract

Large Language Model (LLM) agentic systems solve complex tasks through coordinated workflows, but designing them is a fragile, labor-intensive process. The **Agentic Supernet** paradigm automates this by optimizing a probabilistic space of architectures. However, its reliance on *absolute* performance scores creates a critical flaw: the learning signal entangles an architecture's intrinsic merit with the extrinsic difficulty of the evaluation query. This entanglement leads to unstable search, where simple queries misleadingly inflate weak designs and difficult queries unfairly suppress strong ones. To resolve this, we introduce **RAAS** (Relative Architecture Adaptive Search), a framework that disentangles architectural quality from problem difficulty. Instead of relying on noisy absolute scores, RAAS evaluates a cohort of candidate architectures head-to-head on the *same query*. By learning from their **relative advantage**, it synthesizes a stable, context-fair learning signal that isolates true architectural superiority. This intra-group, relative assessment provides clear and consistent guidance for the search process. Extensive experiments across six benchmarks show that RAAS not only discovers significantly more performant architectures—improving HumanEval pass@1 from 92.23% to 96.31% and MATH accuracy from 52.08% to 60.87%—but also does so with greater sample efficiency and stability, demonstrating that disentangled, relative evaluation is key to robust agentic architecture search.

Introduction

Large Language Models (LLMs) have unlocked new frontiers in complex problem-solving through agentic systems, where collaborative agents tackle tasks beyond the capability of any single model (Brown et al. 2020; OpenAI 2023; Russell and Norvig 2020). Frameworks such as AutoGen (Wu et al. 2024) and CAMEL (Li et al. 2023) provide structural scaffolding for multi-agent coordination, enabling diverse reasoning, planning, and tool-usage workflows. However, the performance of these systems fundamentally depends on the underlying **agentic architecture**—the configuration of agents, their roles, and the communication topology that governs their interaction.

As the complexity of tasks increases, manually designing optimal agentic architectures becomes increasingly infeasible, motivating efforts to **automate architecture discovery**. A recent paradigm shift is the **Agentic Supernet** (Zhang et al. 2025a), which optimizes over a probabilistic space of architectures instead of a single fixed structure. This reformulation enables dynamic generation of task-specific workflows but also inherits a critical flaw from prior work: **learning from absolute performance**. When the score of an architecture depends on both its intrinsic quality and the extrinsic difficulty of a query, these two factors become entangled—corrupting the optimization signal.

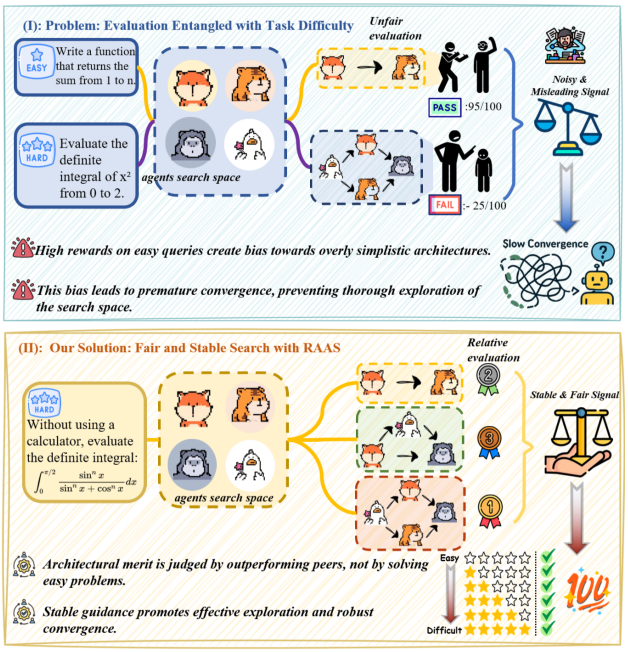

Figure 1: **Problem and Proposed Solution Overview.** (A) Absolute-score evaluation entangles architectural quality with query difficulty, destabilizing search. (B) **RAAS** evaluates cohorts on the same query and learns from relative advantage, yielding fair, stable signals.

How can we evaluate an agentic system's merit independently of task difficulty, and how can we guide the search toward genuinely superior architectures?

*These authors contributed equally.

Copyright © 2026, Trustworthy Agentic AI Workshop@ Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

This entanglement leads to systematic failure in architecture search. Easy queries yield deceptively strong signals for weak architectures, while hard queries suppress genuinely capable ones. As optimization proceeds, the search process reinforces accidental correlations with query difficulty rather than genuine architectural superiority, resulting in brittle, underperforming systems.

To resolve this issue, we propose the **Relative Architecture Adaptive Search (RAAS)** framework. RAAS fundamentally reframes evaluation from *absolute* performance to *relative* advantage. Instead of asking "How well did this architecture perform?", RAAS asks "How much better or worse did this architecture perform relative to its peers on the same query?" By evaluating cohorts of architectures head-to-head, RAAS isolates architectural merit from task difficulty, yielding a zero-centered and stable advantage signal that guides the search reliably. Figure 1 illustrates this contrast: while conventional methods are distorted by evaluation entanglement, RAAS achieves fair and stable optimization through relative, group-wise comparisons.

Our contributions are summarized as follows:

❶ ***Problem Formulation.*** We are the first to formalize the entanglement between architectural quality and task difficulty as a fundamental source of instability in agentic supernet optimization.

❷ ***Methodological Innovation.*** We introduce **RAAS**, which learns from relative advantage signals to achieve stable, difficulty-invariant evaluation and efficient architecture discovery.

❸ ***Empirical Validation.*** Extensive experiments demonstrate that RAAS consistently discovers stronger and more efficient architectures than state-of-the-art baselines across multiple reasoning and planning benchmarks.

LLM Agents and Agentic Systems

The rapid progress of Large Language Models (LLMs) has enabled the creation of powerful autonomous agents capable of complex reasoning, planning, and tool use (Shen et al. 2023; Zhu et al. 2024). However, the performance of a single agent remains inherently limited. Recent research demonstrates that organizing multiple agents into a **Multi-Agent System (MAS)** can substantially enhance problem-solving capacity through structured communication and collaboration (Wang et al. 2024).

Early representative works, including AutoGen (Wu et al. 2024), LLM-Debate (Du et al. 2024), and AgentVerse (Chen et al. 2023b), explored role-based and debate-style collaborations, confirming the potential of collective intelligence. Yet, these systems depend heavily on handcrafted role assignments and communication schemas, demanding significant domain expertise and limiting scalability. This motivates the need for **automated design of agentic architectures** that can adapt to varying task environments.

Automation of Agentic Workflows

To reduce manual effort and improve adaptability, automation of agentic workflows has become a key research direction. Prior works can be broadly categorized into three lines: **(I) Prompt Optimization**, where methods like PromptBreeder (Fernando et al. 2023) and DsPy (Khattab et al. 2023) automatically evolve prompts; **(II) Communication Optimization**, where frameworks such as GPTSwarm (Zhuge et al. 2024) and DyLAN (Liu et al. 2024) refine interaction graphs; and **(III) Role/Profile Evolution**, where approaches such as EvoAgent (Yuan et al. 2024) and AutoAgents (Chen et al. 2023a) use evolutionary search to design agent behaviors.

More recent systems pursue full automation: ADAS (Hu, Lu, and Clune 2025), AgentSquare (Shang et al. 2025), and AFlow (Zhang et al. 2025b) automatically synthesize entire workflows across vast search spaces. Our direct baseline, `MaAS` (Zhang et al. 2025a), introduces the **Agentic Supernet** paradigm—optimizing distributions of architectures rather than single configurations—to generate query-specific workflows dynamically. Building upon this foundation, our work targets the core instability of `MaAS`: its reliance on absolute performance signals.

Robust Learning Signals

Automated agentic design parallels challenges in **Neural Architecture Search (NAS)** (Ren et al. 2021), where reliable optimization signals are essential for discovering performant designs. Existing methods typically depend on *absolute task scores*, which entangle architectural quality with query difficulty and cause unstable learning dynamics.

Recent studies propose *relative, peer-referenced evaluation*, where systems are compared within query-specific cohorts rather than against global baselines. Examples include sequence-level optimization in SCST (Rennie et al. 2017) and grouped ranking approaches like GRPO (Shao et al. 2024), both emphasizing fair, on-task comparison. Inspired by these insights, our **RAAS** framework integrates *grouped, per-query relative evaluation* into the Agentic Supernet paradigm, producing stable and efficient search dynamics without requiring additional critics or complex reward models.

Design Principles. To achieve stability and reliability in automated agentic search, RAAS is grounded in three core design principles:

Principles of Robust Agentic Supernet Design

❶ **Optimality:** *Discover architectures that maximize overall utility by balancing performance and computational cost.*

❷ **Stability:** *Ensure reliable optimization by leveraging relative comparisons instead of volatile absolute scores.*

❸ **Efficiency:** *Accelerate discovery of high-quality architectures through consistent, low-variance learning signals.*

In the subsequent sections, we demonstrate how `RAAS` adheres to these principles to achieve stable, fair, and efficient search through the formulation of Relative Advantage Synthesis.

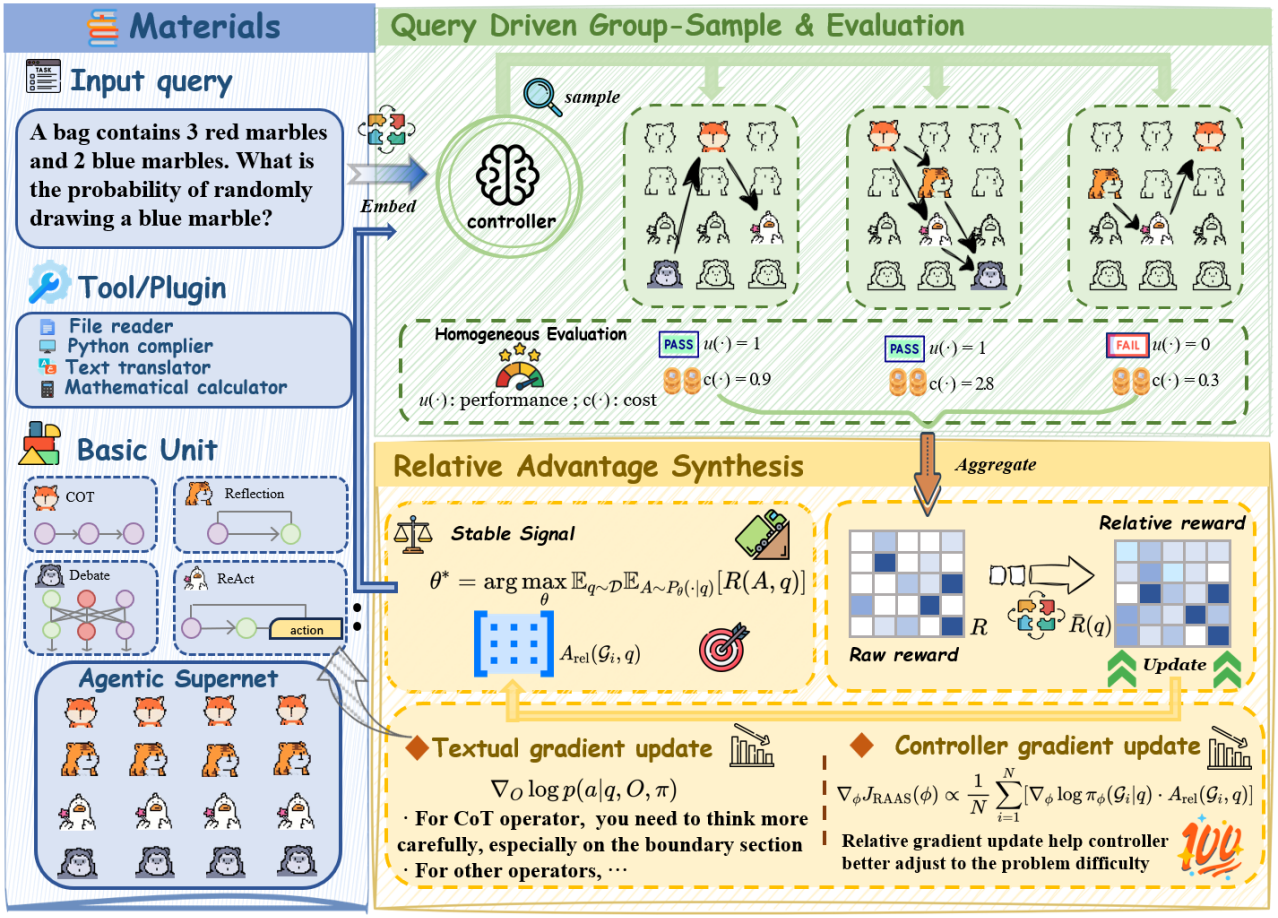

Figure 2: **RAAS overview.** For each query, RAAS samples a cohort from the agentic supernet, computes a peer-group baseline, synthesizes zero-centered relative advantages, and applies advantage-weighted updates. Two lightweight controls (R, τ) modulate exploration and early stopping.

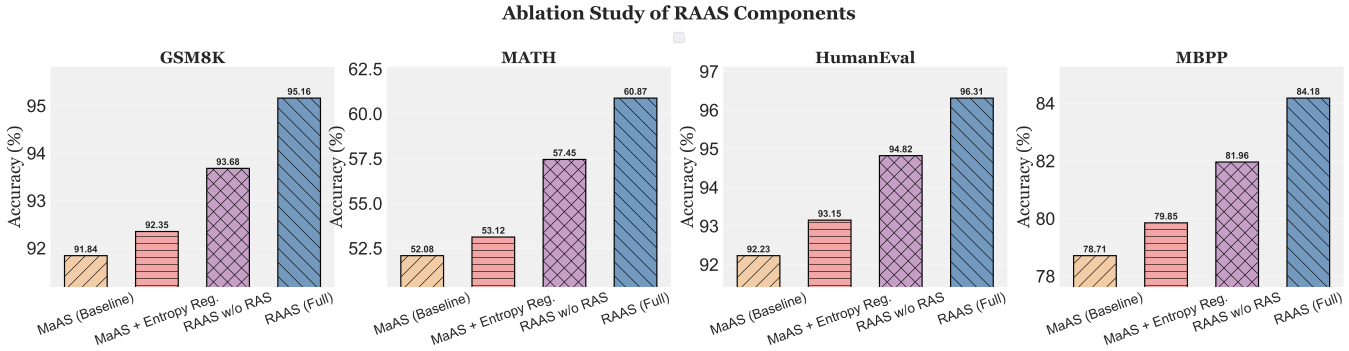

Figure 3: **Ablation studies.** Impact of key components on accuracy and stability across domains. Relative normalization and advantage-weighted updates are most critical; controls (R, τ) improve efficiency without hurting accuracy.

Methodology

RAAS: Relative Architecture Adaptive Search

In this section we present RAAS, a system-level optimization framework for discovering effective agentic workflows within the supernet paradigm. Unlike methods that score each sampled architecture against *absolute* metrics, RAAS evaluates *cohorts* of designs on the *same* query and reasons about their *relative* merit, yielding stable, difficulty-invariant learning signals.

Table 1: **Main results across agentic systems.** Accuracy (%) on multiple benchmarks. Best and second-best are **bold** and underlined.

Method	Backbone: gpt-4o-mini					Backbone: qwen-2.5-70b				
	MATH	GSM8K	HumanEval	MBPP	Average	MATH	GSM8K	HumanEval	MBPP	Average
Vanilla	46.30	87.45	87.08	71.83	73.16	43.29	83.22	84.03	69.85	70.10
CoT	46.89 ↑0.59	87.28 ↓0.17	88.52 ↑1.44	74.21 ↑2.38	74.23 ↑1.07	45.83 ↑2.54	86.91 ↑3.69	87.82 ↑3.79	72.96 ↑3.11	73.38 ↑3.28
ComplexCoT	46.91 ↑0.61	87.18 ↓0.27	87.82 ↑0.74	73.95 ↑2.12	73.97 ↑0.81	45.58 ↑2.29	86.77 ↑3.55	87.21 ↑3.18	72.83 ↑2.98	73.10 ↑3.00
Self-Consistency	48.47 ↑2.17	87.78 ↑0.33	89.12 ↑2.04	76.18 ↑4.35	75.39 ↑2.23	47.18 ↑3.89	87.27 ↑4.05	88.57 ↑4.54	75.12 ↑5.27	74.54 ↑4.44
MultiPersona	45.74 ↓0.56	87.69 ↑0.24	88.83 ↑1.75	73.61 ↑1.78	73.97 ↑0.81	44.68 ↑1.39	86.95 ↑3.73	87.74 ↑3.71	72.58 ↑2.73	72.99 ↑2.89
LLM-Debate	48.85 ↑2.55	89.17 ↑1.72	88.76 ↑1.68	75.63 ↑3.80	75.60 ↑2.44	47.94 ↑4.65	88.51 ↑5.29	88.19 ↑4.16	74.87 ↑5.02	74.88 ↑4.78
LLM-Blender	47.16 ↑0.86	88.47 ↑1.02	88.93 ↑1.85	74.72 ↑2.89	74.82 ↑1.66	46.21 ↑2.92	87.84 ↑4.62	88.41 ↑4.38	74.09 ↑4.24	74.14 ↑4.04
DyLAN	48.97 ↑2.67	90.18 ↑2.73	90.54 ↑3.46	76.52 ↑4.69	76.55 ↑3.39	48.41 ↑5.12	89.76 ↑6.54	90.08 ↑6.05	75.94 ↑6.09	76.05 ↑5.95
AgentVerse	47.56 ↑1.26	89.92 ↑2.47	89.31 ↑2.23	75.64 ↑3.81	75.61 ↑2.45	46.73 ↑3.44	89.45 ↑6.23	88.71 ↑4.68	75.03 ↑5.18	74.98 ↑4.88
MacNet	45.27 ↓1.03	87.96 ↑0.51	84.64 ↓2.44	72.59 ↑0.76	72.62 ↓0.54	44.52 ↑1.23	87.48 ↑4.26	84.02 ↓0.01	71.97 ↑2.12	71.99 ↑1.89
AutoAgents	45.43 ↓0.87	87.73 ↑0.28	87.71 ↑0.63	71.94 ↑0.11	73.20 ↑0.04	44.67 ↑1.38	87.29 ↑4.07	87.18 ↑3.15	71.48 ↑1.63	72.66 ↑2.56
GPTSwarm	48.17 ↑1.87	89.26 ↑1.81	89.43 ↑2.35	75.61 ↑3.78	75.62 ↑2.46	47.52 ↑4.23	88.87 ↑5.65	89.02 ↑4.99	75.14 ↑5.29	75.14 ↑5.04
ADAS	43.47 ↓2.83	86.28 ↓1.17	84.36 ↓2.72	71.38 ↓0.45	71.37 ↓1.79	42.84 ↓0.45	85.89 ↑2.67	83.94 ↓0.09	70.91 ↑1.06	70.90 ↑0.80
AgentSquare	48.76 ↑2.46	87.78 ↑0.33	89.25 ↑2.17	75.27 ↑3.44	75.27 ↑2.11	48.09 ↑4.80	87.34 ↑4.12	88.81 ↑4.78	74.79 ↑4.94	74.76 ↑4.66
AFlow	51.43 ↑5.13	91.27 ↑3.82	91.04 ↑3.96	77.91 ↑6.08	77.91 ↑4.75	50.79 ↑7.50	90.87 ↑7.65	90.71 ↑6.68	77.46 ↑7.61	77.46 ↑7.36
MaAS	52.08 ↑5.78	91.84 ↑4.39	92.23 ↑5.15	78.71 ↑6.88	78.72 ↑5.56	51.49 ↑8.20	91.42 ↑8.20	91.91 ↑7.88	78.27 ↑8.42	78.27 ↑8.17
RAAS (Ours)	**60.87** ↑14.57	**95.16** ↑7.71	**96.31** ↑9.23	**84.18** ↑12.35	**84.13** ↑10.97	**60.14** ↑16.85	**94.69** ↑11.47	**95.96** ↑11.93	**83.59** ↑13.74	**83.59** ↑13.49

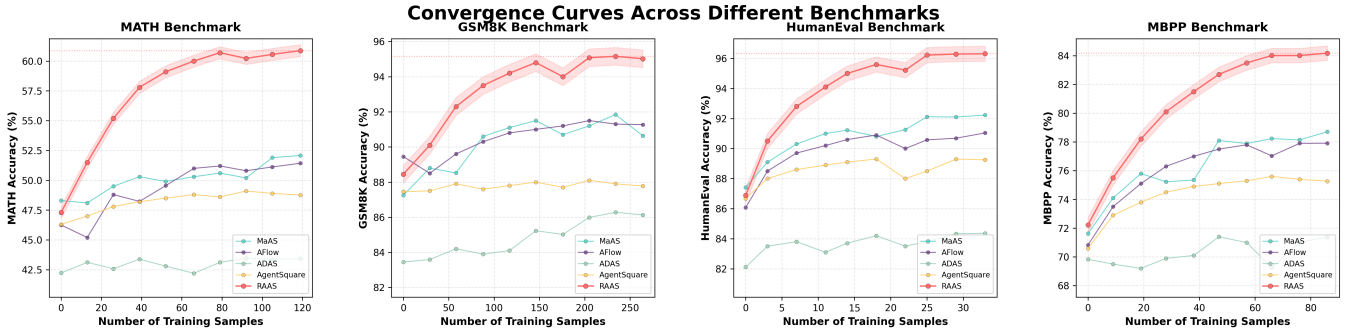

Figure 4: **Convergence comparison across benchmarks.** RAAS converges faster and more stably than baselines across all domains. The red curve (RAAS) reaches higher performance with fewer steps and reduced variance (confidence bands).

Agentic Supernet Foundation

Following the supernet formulation (Zhang et al. 2025a), we define the **Agentic Supernet** $\mathcal{A} = \{\pi, \mathbb{O}\}$ as a probabilistic composition of operators over L layers:

$$\mathcal{A} = \left\{ \{\pi_l(\mathcal{O})\}_{\mathcal{O}\in\mathbb{O}} \right\}_{l=1}^{L}, \quad \pi_l(\mathcal{O}) = p(\mathcal{O} \mid \mathcal{A}_{1:l-1}). \quad (1)$$

A concrete architecture $\mathcal{G}$ activates operator sets $\mathcal{V}_l$ layer-wise with joint probability

$$p(\mathcal{G}) = \prod_{l=1}^{L} \prod_{\mathcal{O}\in\mathbb{O}} \pi_l(\mathcal{O})^{\mathbb{I}_{\mathcal{O}\in\mathcal{V}_l}}. \quad (2)$$

We seek a query-conditioned distribution $\mathbb{P}(\mathcal{G} \mid q)$ that maximizes cost-adjusted utility:

$$\max_{\mathbb{P}(\mathcal{G}|q)} \ \mathbb{E}_{\mathcal{D}}[U_\lambda(\mathcal{G}; q)] \ \text{ s.t. } \mathcal{G} \subset \mathcal{A}, \quad U_\lambda = U - \lambda C. \quad (3)$$

The central difficulty is *evaluation*: absolute scores $R(\mathcal{G}, q)$ conflate architectural quality with query difficulty, destabilizing optimization (easy queries inflate weak designs; hard queries depress strong ones).

Core Principle: Relative Advantage

RAAS replaces absolute scoring with *peer-normalized* comparison. For query q, instead of "How good is $\mathcal{G}_i$?", we ask

Method	Level 1	Level 2	Level 3	Average
GPT-4o-mini	7.53	4.40	0.00	3.98
GPT-4	9.85 ↑2.32	2.12 ↓2.28	2.31 ↑2.31	4.76 ↑0.78
AutoGPT	13.54 ↑6.01	0.00 ↓4.40	4.12 ↑4.12	5.89 ↑1.91
TapeAgent	24.12 ↑16.59	15.02 ↑10.62	10.68 ↑10.68	16.61 ↑12.63
Sibyl	22.08 ↑14.55	16.31 ↑11.91	4.42 ↑4.42	14.27 ↑10.29
AutoAgents	16.67 ↑9.14	0.00 ↓4.40	0.00	5.56 ↑1.58
GPTSwarm	24.38 ↑16.85	17.12 ↑12.72	2.27 ↑2.27	14.59 ↑10.61
ADAS	14.42 ↑6.89	4.68 ↑0.28	0.00	6.37 ↑2.39
AgentSquare	23.25 ↑15.72	16.44 ↑12.04	6.78 ↑6.78	15.49 ↑11.51
AFlow	10.75 ↑3.22	8.81 ↑4.41	4.08 ↑4.08	8.00 ↑3.35
MaAS	25.91 ↑18.38	22.01 ↑17.61	6.25 ↑6.25	18.06 ↑13.41
RAAS (Ours)	**29.53** ↑22.00	**25.32** ↑20.92	**7.68** ↑7.68	**20.84** ↑16.86

Table 2: **GAIA results.** Accuracy (%) by difficulty. Best and second-best are **bold** and underlined.

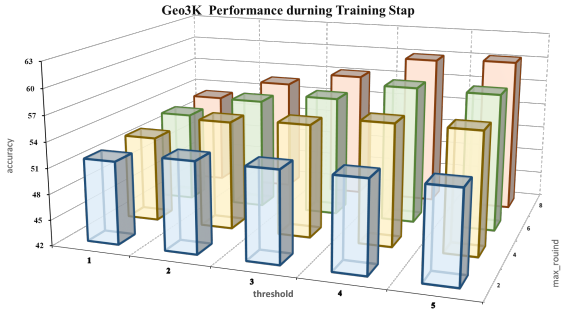

Figure 5: Impact of max sampling rounds R and success threshold τ on efficiency and final performance. Larger R improves discovery on hard queries; moderate τ avoids over-exploration on easy ones.

"How much better or worse is $\mathcal{G}_i$ than its cohort on q?". As shown in Fig. 2, RAAS samples a cohort, executes all candidates on the *same* query, constructs a peer baseline, computes zero-centered advantages, and applies advantage-weighted updates. Two simple controls—maximum rounds R and success threshold τ—adjust exploration breadth and early stopping without altering the core mechanism.

Mechanism: Relative Advantage Synthesis (RAS)

Cohort Evaluation. For each query q, sample N workflows $\mathcal{C}_q = \{\mathcal{G}_1, \ldots, \mathcal{G}_N\}$ and obtain scores $R(\mathcal{G}_i, q)$. Define the *peer baseline*:

$$\bar{R}(q) = \frac{1}{N} \sum_{i=1}^{N} R(\mathcal{G}_i, q). \tag{4}$$

Let $R_i = R(\mathcal{G}_i, q)$ and $A_i = A_{\text{rel}}(\mathcal{G}_i, q)$. Variability decomposes as

$$\text{Var}[R_i] = \text{Var}[\bar{R}(q)] + \text{Var}[A_i] + 2\,\text{Cov}(\bar{R}(q), A_i), \tag{5}$$

where $\text{Var}[\bar{R}(q)]$ captures query difficulty (removed by normalization), and $\text{Var}[A_i]$ isolates architectural differences.

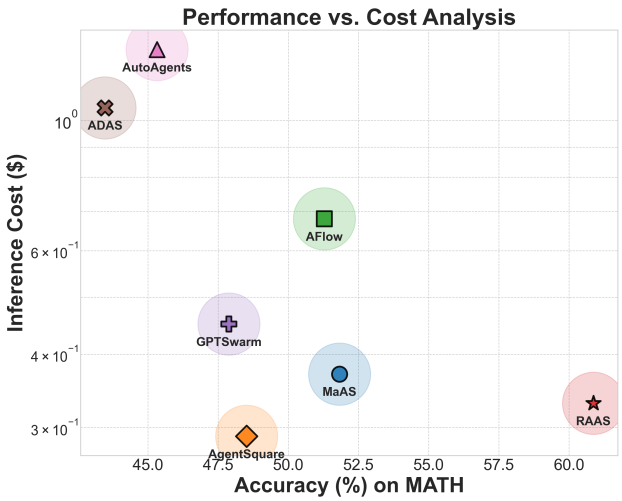

Figure 6: **Cost–performance analysis.** RAAS achieves superior accuracy under equal or reduced compute budgets compared to automated and hand-crafted baselines.

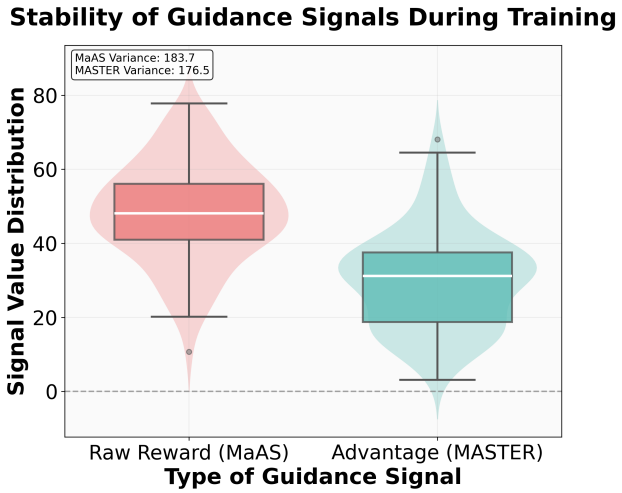

Figure 7: **Stability of learning signal.** Compared to absolute-score optimization, RAAS delivers lower-variance signals and smoother progress across steps, mitigating difficulty-induced oscillation.

Advantage Synthesis. Define the *relative advantage*

$$A_{\text{rel}}(\mathcal{G}_i, q) = R(\mathcal{G}_i, q) - \bar{R}(q), \tag{6}$$

a zero-centered signal independent of difficulty. The supernet controller with parameters ϕ updates via advantage weighting:

$$\nabla_\phi J_{\text{RAAS}}(\phi; q) = \frac{1}{N} \sum_{i=1}^{N} g_i(\phi)\, A_{\text{rel}}(\mathcal{G}_i, q), \tag{7}$$

where $g_i(\phi) = \sum_{t=0}^{T-1} \nabla_\phi \log \pi_\phi(a_{i,t} \mid s_{i,t})$ accumulates architectural decision gradients. Positive A_{rel} reinforces effective patterns; negative A_{rel} downweights weak ones, stabilizing discovery by filtering query-induced variance.

System-Level Controls

To ensure practicality, RAAS exposes two lightweight controls. **Max sampling rounds** R sets exploration breadth. For each query, sample $\{\mathcal{G}^{(1)}, \ldots, \mathcal{G}^{(R)}\}$ and select

$$r^\star = \arg\max_{1 \le r \le R} A_{\text{rel}}(\mathcal{G}^{(r)}, q), \quad \mathcal{G}_q^\star = \mathcal{G}^{(r^\star)}. \tag{8}$$

This increases the chance of discovering consistently positive-advantage designs. **Success threshold** τ enables adaptive early stopping when recent candidates repeatedly achieve nonnegative advantage:

$$c_r = \sum_{j=r-\tau+1}^{r} \mathbb{1}\Big\{ A_{\text{rel}}(\mathcal{G}^{(j)}, q) \ge 0 \Big\} \ \ge \ \tau. \tag{9}$$

This saves computation on easy queries while preserving deeper exploration on hard ones, improving efficiency without compromising RAAS's core relative-evaluation principle.

Experiments

Experimental Setup

Tasks and Benchmarks. We evaluate RAAS on three complementary domains to reflect practical deployment: **(1) math reasoning**—GSM8K (Cobbe et al. 2021), MATH (Hendrycks et al. 2021), MultiArith (Roy and Roth 2016); **(2) code generation**—HumanEval (Chen et al. 2021), MBPP (Austin et al. 2021); and **(3) complex tool use**—GAIA (Mialon et al. 2024). We follow standard splits and evaluation protocols consistent with prior work (Zhang et al. 2025a) for fair comparison.

Baselines. We compare against **(i) single-agent** methods (CoT (Wei et al. 2023), Self-Consistency (Wang et al. 2023)); **(ii) hand-crafted multi-agent** systems (LLM-Debate (Du et al. 2024), AgentVerse (Chen et al. 2023b)); and **(iii) automated agentic** systems (GPTSwarm (Zhuge et al. 2024), AgentSquare (Shang et al. 2025), AFlow (Zhang et al. 2025b), MaAS (Zhang et al. 2025a)).

Implementation Details. Following the MaAS setup, we adopt the same multi-domain arrangement covering math reasoning, coding, and knowledge Q&A to optimize supernet parameters. All experiments use `gpt-4o-mini` and `qwen-2.5-70b` as underlying LLMs.

Superiority (Q1)

We report cross-domain results in Table 1. RAAS consistently surpasses automated baselines and hand-crafted systems across all benchmarks.

Headline results. RAAS attains **60.87% on MATH** (+8.79 over MaAS), **95.16% on GSM8K** (+3.32), **96.31% on HumanEval** (+4.08), and **84.18% on MBPP** (+5.47), yielding an **average +5.41 points**. On GAIA (Table 2), RAAS advances the SOTA at all difficulty levels, reaching **20.84% average** (+2.78 over MaAS) with comparable compute.

Cross-domain generalization. Obs. ❶ *Math reasoning:* +8.79 on MATH ($\sim$16.9% relative) indicates effectiveness under high difficulty variance. **Obs. ❷** *Code generation:* Gains on HumanEval (+4.08) and MBPP (+5.47) show transfer beyond reasoning to structured generation where syntactic and semantic correctness must be jointly optimized. **Obs. ❸** *Tool use:* On GAIA, improvements across all difficulty levels confirm that RAAS handles multi-step planning and tool invocation where agent coordination is critical.

Ablation Studies

We ablate key components of RAAS (cohort size, peer baseline, advantage-weighted update, and controls R, τ). As shown in Fig. 3, each component contributes; removing relative normalization or advantage weighting notably degrades both stability and final accuracy.

Cost-Performance (Q3)

We assess the compute–accuracy trade-off in Fig. 6. Under comparable or lower cost, RAAS reaches higher accuracy than baselines, reflecting more sample-efficient search guided by relative advantages.

Stability (Q2)

Relative advantage reduces variance by normalizing out query difficulty. We visualize signal stability across training in Fig. 7: RAAS exhibits smoother trajectories with narrower bands, avoiding premature convergence traps observed in absolute-reward methods.

Conclusion

Our work addresses evaluation signal entanglement in agentic architecture search—where absolute scores conflate architectural merit with query difficulty. We reframe automated multi-agent design from pursuing point-wise accuracy to seeking comparative advantage through peer evaluation. **RAAS** evaluates architectures head-to-head on identical queries, selecting designs by relative advantage, while *Relative Advantage Synthesis* aggregates peer comparisons to provide stable, query-normalized learning signals. This gradient-free approach adds only linear overhead in cohort size while maintaining efficiency. Across mathematical reasoning, code generation, and tool-use tasks, RAAS consistently achieves 5.41 points average improvement while remaining computationally practical. Future work includes extending relative advantage to multi-turn interactions, designing task-aware cohort sampling strategies, and deepening theory connecting peer evaluation with generalization.

References

Austin, J.; Odena, A.; Nye, M. I.; Bosma, M.; Michalewski, H.; Dohan, D.; Jiang, E.; Cai, C. J.; Terry, M.; Le, Q. V.; and Sutton, C. 2021. Program Synthesis with Large Language Models. *CoRR*, abs/2108.07732.

Brown, T. B.; Mann, B.; Ryder, N.; et al. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165.

Chen, G.; Dong, S.; Shu, Y.; Zhang, G.; Sesay, J.; Karlsson, B. F.; Fu, J.; and Shi, Y. 2023a. AutoAgents: A Framework for Automatic Agent Generation. *CoRR*, abs/2309.17288.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; de Oliveira Pinto, H. P.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; Ray, A.; Puri, R.; Krueger, G.; Petrov, M.; Khlaaf, H.; Sastry, G.; Mishkin, P.; Chan, B.; Gray, S.; Ryder, N.; Pavlov, M.; Power, A.; Kaiser, L.; Bavarian, M.; Winter, C.; Tillet, P.; Such, F. P.; Cummings, D.; Plappert, M.; Chantzis, F.; Barnes, E.; Herbert-Voss, A.; Guss, W. H.; Nichol, A.; Paino, A.; Tezak, N.; Tang, J.; Babuschkin, I.; Balaji, S.; Jain, S.; Saunders, W.; Hesse, C.; Carr, A. N.; Leike, J.; Achiam, J.; Misra, V.; Morikawa, E.; Radford, A.; Knight, M.; Brundage, M.; Murati, M.; Mayer, K.; Welinder, P.; McGrew, B.; Amodei, D.; McCandlish, S.; Sutskever, I.; and Zaremba, W. 2021. Evaluating Large Language Models Trained on Code. *CoRR*, abs/2107.03374.

Chen, W.; Su, Y.; Zuo, J.; Yang, C.; Yuan, C.; Qian, C.; Chan, C.-M.; Qin, Y.; Lu, Y.; Xie, R.; Liu, Z.; Sun, M.; and Zhou, J. 2023b. AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors in Agents. *CoRR*, abs/2308.10848.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *CoRR*, abs/2110.14168.

Du, Y.; Li, S.; Torralba, A.; Tenenbaum, J. B.; and Mordatch, I. 2024. Improving Factuality and Reasoning in Language Models through Multiagent Debate.

Fernando, C.; Banarse, D.; Michalewski, H.; Osindero, S.; and Rocktäschel, T. 2023. Promptbreeder: Self-Referential Self-Improvement Via Prompt Evolution. *CoRR*, abs/2309.16797.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Hu, S.; Lu, C.; and Clune, J. 2025. Automated Design of Agentic Systems. arXiv:2408.08435.

Khattab, O.; et al. 2023. DSPy: Compiling Declarative Language Model Calls into Self-Improving Pipelines. arXiv:2310.03714.

Li, G.; Hammoud, H. A. A. K.; Itani, H.; Khizbullin, D.; and Ghanem, B. 2023. CAMEL: Communicative Agents for "Mind" Exploration of Large Language Model Society. arXiv:2303.17760.

Liu, Z.; Zhang, Y.; Li, P.; Liu, Y.; and Yang, D. 2024. A Dynamic LLM-Powered Agent Network for Task-Oriented Agent Collaboration. arXiv:2310.02170.

Mialon, G.; Fourrier, C.; Wolf, T.; LeCun, Y.; and Scialom, T. 2024. GAIA: a benchmark for General AI Assistants. In *The Twelfth International Conference on Learning Representations*.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Ren, P.; Xiao, Y.; Chang, X.; Huang, P.-Y.; Li, Z.; Chen, X.; and Wang, X. 2021. A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions. arXiv:2006.02903.

Rennie, S. J.; Marcheret, E.; Mroueh, Y.; Ross, J.; and Goel, V. 2017. Self-critical Sequence Training for Image Captioning. arXiv:1612.00563.

Roy, S.; and Roth, D. 2016. Solving General Arithmetic Word Problems. *CoRR*, abs/1608.01413.

Russell, S. J.; and Norvig, P. 2020. *Artificial Intelligence: A Modern Approach (4th Edition)*. Pearson. ISBN 9781292401133.

Shang, Y.; Li, Y.; Zhao, K.; Ma, L.; Liu, J.; Xu, F.; and Li, Y. 2025. AgentSquare: Automatic LLM Agent Search in Modular Design Space. arXiv:2410.06153.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y. K.; Wu, Y.; and Guo, D. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv:2402.03300.

Shen, Y.; Song, K.; Tan, X.; Li, D.; Lu, W.; and Zhuang, Y. 2023. HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in HuggingFace. *CoRR*, abs/2303.17580.

Wang, L.; Ma, C.; Feng, X.; Zhang, Z.; Yang, H.; Zhang, J.; Chen, Z.; Tang, J.; Chen, X.; Lin, Y.; Zhao, W. X.; Wei, Z.; and Wen, J. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6).

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q. V.; Chi, E. H.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *The Eleventh International Conference on Learning Representations*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; and Zhou, D. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903.

Wu, Q.; Bansal, G.; Zhang, J.; Wu, Y.; Li, B.; Zhu, E.; Jiang, L.; Zhang, X.; Zhang, S.; Liu, J.; Awadallah, A. H.; White, R. W.; Burger, D.; and Wang, C. 2024. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation Framework. In *COLM*.

Yuan, S.; Song, K.; Chen, J.; Tan, X.; Li, D.; and Yang, D. 2024. EvoAgent: Towards Automatic Multi-Agent Generation via Evolutionary Algorithms. *CoRR*, abs/2406.14228.

Zhang, G.; Niu, L.; Fang, J.; Wang, K.; Bai, L.; and Wang, X. 2025a. Multi-agent Architecture Search via Agentic Supernet. arXiv:2502.04180.

Zhang, J.; Xiang, J.; Yu, Z.; Teng, F.; Chen, X.; Chen, J.; Zhuge, M.; Cheng, X.; Hong, S.; Wang, J.; Zheng, B.; Liu, B.; Luo, Y.; and Wu, C. 2025b. AFlow: Automating Agentic Workflow Generation. arXiv:2410.10762.

Zhu, Y.; Qiao, S.; Ou, Y.; Deng, S.; Zhang, N.; Lyu, S.; Shen, Y.; Liang, L.; Gu, J.; and Chen, H. 2024. KnowAgent: Knowledge-Augmented Planning for LLM-Based Agents. *CoRR*, abs/2403.03101.

Zhuge, M.; Wang, W.; Kirsch, L.; Faccio, F.; Khizbullin, D.; and Schmidhuber, J. 2024. GPTSwarm: Language Agents as Optimizable Graphs. In *Forty-first International Conference on Machine Learning*.

Reproducibility Checklist

This paper:

- Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) **yes**
- Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) **yes**
- Provides well marked pedagogical references for less-familiare readers to gain background necessary to replicate the paper (yes/no) **yes**

Does this paper make theoretical contributions? (yes/no) yes

If yes, please complete the list below.

- All assumptions and restrictions are stated clearly and formally. (yes/partial/no) **yes**
- All novel claims are stated formally (e.g., in theorem statements). (yes/partial/no) **yes**
- Proofs of all novel claims are included. (yes/partial/no) **yes**
- Proof sketches or intuitions are given for complex and/or novel results. (yes/partial/no) **yes**
- Appropriate citations to theoretical tools used are given. (yes/partial/no) **yes**
- All theoretical claims are demonstrated empirically to hold. (yes/partial/no/NA) **yes**
- All experimental code used to eliminate or disprove claims is included. (yes/no/NA) **yes**

Does this paper rely on one or more datasets? (yes/no) yes

If yes, please complete the list below.

- A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) **yes**
- All novel datasets introduced in this paper are included in a data appendix. (yes/partial/no/NA) **yes**
- All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. (yes/partial/no/NA) **yes**
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations. (yes/no/NA) **yes**
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available. (yes/partial/no/NA) **yes**
- All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing. (yes/partial/no/NA) **yes**

Does this paper include computational experiments? (yes/no) yes

If yes, please complete the list below.

- This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting. (yes/partial/no/NA) **no**
- Any code required for pre-processing data is included in the appendix. (yes/partial/no) **no**
- All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no) **partial**
- All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. (yes/partial/no) **yes**
- All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no) **yes**
- If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results. (yes/partial/no/NA) **yes**
- This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks. (yes/partial/no) **yes**
- This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics. (yes/partial/no) **yes**
- This paper states the number of algorithm runs used to compute each reported result. (yes/no) **yes**
- Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information. (yes/no) **yes**
- The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank). (yes/partial/no) **yes**
- This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments. (yes/partial/no/NA) **yes**