Breaking the Evaluation Paradox: Evaluating High-Entropy Search with Computationally Irreducible Constraints

Juntao Wu[*] [†] [12], **Wei Wen**[*] [2], **Xianting Huang** [3], **Shuai Pang** [1], **Ruizhi Qiao** [2], **Xing Sun** [2], **Ke Wang**[‡] [1]

[1] Jinan University, [2] Tencent Youtu Lab, [3] The Chinese University of Hong Kong

{jtitor, wangke}@jnu.edu.cn, ps2023104429@stu2023.jnu.edu.cn,

{jawnrwen, ruizhiqiao, winfredsun}@tencent.com, alisonwww@link.cuhk.edu.hk

Abstract

Evaluating the exhaustive search capabilities of large language models (LLMs) is plagued by a fundamental paradox: verifying completeness requires complete ground truth, yet high-entropy enumeration tasks make such ground truth impossible for humans to create. This causes benchmarks to systematically penalize models for outperforming their human annotators. Despite rapid progress in web-search and deep research agents—which now issue hundreds of queries, traverse diverse sites, and synthesize long reports—evaluation still largely relies on partially annotated answer sets, LLM-based judges, or single-answer questions that avoid genuinely exhaustive search scenarios.

We break this paradox by shifting the evaluation paradigm from simulating a messy reality to constructing computationally pure challenges. We introduce VERITAS (Verifiable Traversal Assessment for Search), a framework built on the principle of **computationally irreducible constraints**. By introducing novel, non-optimizable constraints, we create verifiable, sparse-answer search tasks that are computationally equivalent to exhaustive enumeration. These constraints are easy to verify but impossible for LLMs or search engines to optimize, forcing agents to genuinely traverse the entire search space. VERITAS can automatically generate a virtually infinite number of test cases with perfect ground truth and precise difficulty control, with marginal instance cost dominated by hash computations. This provides not only a robust benchmark for evaluating systematic exploration under uncertainty but also a scalable method for generating training data to improve these crucial, yet underdeveloped, capabilities.

1 Introduction

Large language models (LLMs) achieve strong performance on factual question-answering benchmarks with single, deterministic targets (Kwiatkowski et al., 2019; Joshi et al., 2017), but their behavior on high-entropy queries (HEQs)—tasks that require enumerating large open-ended answer sets—remains underexplored. Many real-world information needs resemble HEQs: success depends on recall over a broad candidate space rather than on one "best" answer, and agents must decide when additional exploration is no longer worthwhile.

Recent web-search and "deep research" agents push LLMs directly into this regime: commercial and open-source systems routinely issue hundreds of queries, traverse diverse sites, and synthesize long reports from heterogeneous evidence (OpenAI, 2025; Perplexity, 2025; Wei et al., 2025; Tao et al.; Li et al., 2025b). Benchmarks such as GAIA and BrowseComp (Mialon et al., 2023; Wei et al., 2025; Zhou et al., 2025; Wei et al.) evaluate multi-step browsing on "hard to find but easy to verify" questions, while WideSearch and DeepWideSearch (Wong et al., 2025; Lan et al., 2025) target breadth and depth of coverage over thousands of entities. With the growing attention to complex information needs in real-world scenarios, however, the majority of these benchmarks rely on partially annotated answer pools, which not only incur considerable costs but also risk penalizing agents that discover valid items beyond the pool as hallucinating.

We call this the *evaluation paradox* for high-entropy search: exhaustive search should be judged against a complete ground-truth set, yet for HEQs such ground truth is practically intractable to obtain. The same open-endedness that makes these tasks hard for models prevents annotators from certifying completeness, so annotated answer pools are al-

[*] Equal Contribution.

[†] Work done during internship at Tencent.

[‡] Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 28209–28218

most surely incomplete. Under standard evaluation protocols, any correct item outside the annotated set is indistinguishable from an error, and systems that discover more valid items can be penalized as hallucinating. As a result, measured performance is effectively bounded by annotation coverage rather than by an agent's true search capability, even when wide-search benchmarks attempt to increase coverage with large manually curated pools (Wong et al., 2025; Lan et al., 2025).

In this work we take a complementary approach. Instead of approximating real-world HEQs with partially annotated answer sets, we construct *synthetic objectives* on top of real web-scale domains. Our VERITAS benchmark uses cryptographic hash constraints to turn high-entropy enumeration into sparse-answer search tasks that are *easy to verify but impossible to optimize semantically*. Agents still have to traverse genuine web content (e.g., artist discographies, model hubs) using the same kinds of search and browsing tools as in existing benchmarks, but success reduces to finding items whose identifiers satisfy a non-optimizable hash constraint, enabling exact labels and scalable instance generation.

Our contributions are threefold:

1. We formalize the evaluation paradox for high-entropy search tasks and analyze why it is difficult to resolve within annotation-based paradigms.

2. We introduce VERITAS, a benchmark that uses computationally irreducible, hash-based constraints to construct scalable, automatically verifiable tests of exhaustive search capabilities without human labels or LLM judges.

3. We combine theoretical and empirical analyses to show that hash-constrained sparse search is computationally equivalent to exhaustive enumeration, and we evaluate several state-of-the-art web agents on VERITAS to characterize their failure modes on high-entropy search tasks.

2 Related Work

Evaluation of information-seeking systems has evolved from static QA benchmarks such as HotpotQA and related multi-hop datasets (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022; Press et al., 2022) to dynamic, agent-centric web benchmarks that require navigation and tool use (Mialon et al., 2023; Wei et al., 2025; Zhou et al., 2025; Wei et al.; Wong et al., 2025; Lan et al., 2025). WebArena (Zhou et al., 2023) introduced self-hosted realistic web environments for autonomous agents, establishing foundational evaluation protocols. Subsequently, AgentBench (Liu et al., 2023) evaluates LLM agents in 8 environments, including web browsing. Specifically, GAIA and BrowseComp construct "hard to find but easy to verify" tasks that emphasize multi-step browsing under fuzzy, underspecified instructions, whereas WideSearch and DeepWideSearch explicitly target breadth and depth of search over thousands of entities. All of these benchmarks ultimately rely on large but still incomplete manually annotated answer pools, so coverage limitations remain a central bottleneck.

Commercial "deep research" products such as those from OpenAI, Perplexity, Gemini, Kimi, Doubao, and Grok (OpenAI, 2025; Perplexity, 2025; Gemini, 2025; Kimi, 2025; Doubao, 2025; x.ai, 2025), together with open-source frameworks like WebShaper, WebSailor, WebThinker and WebDancer (Tao et al.; Li et al., 2025a,b; Wu et al., 2025), push LLM agents into long-horizon web settings and typically evaluate them on GAIA, BrowseComp, WideSearch, and related suites (Zhang et al., 2025; Xi et al., 2025). Alongside these explorations and the application of reinforcement learning to search agents (Song et al., 2025; Jin et al., 2025), VERITAS's potential as an important data source correspondingly emerges.

Moreover, recent surveys on LLM-based agents and deep research (Guo et al., 2024; Wang et al., 2024; Zhang et al., 2025; Xi et al., 2025) likewise highlight that high-entropy, partially labeled conditions make it difficult to measure exhaustive search capability. VERITAS is complementary to these efforts: by introducing **computationally irreducible constraints** via cryptographic hash functions, it isolates the enumeration component of deep research and provides exact, automatically scalable supervision rather than relying on manually constructed answer sets.

3 Method

3.1 Evaluation Paradox

Exhaustive search tasks are fundamentally ill-posed for evaluation: verifying completeness requires the very ground truth that the task itself makes intractable to obtain.

The difficulty stems from two inherent properties

Benchmark	Wide Degree	Scalable	Human Depend	Verification Difficulty	Core Issue
GAIA	⋆ Low (Single ans)	No Hard	Yes Expert	Easy Single answer Easy verification	Not Wide Human dependent
BrowseComp (OpenAI 2025)	⋆ Low (Single ans)	No Hard	Yes Full	Easy Single answer Easy verification	Not Wide Human dependent
WideSearch	⋆⋆⋆ Medium (Wide)	No Hard	Yes Full	**Hard** Many entities LLM unstable	Human annotation May punish better answers
VERITAS (Ours)	⋆⋆⋆⋆⋆ **Extreme (Unlimited)**	**Yes** **Easy**	**No** **Zero**	**Easy** **Easy** **verification**	

Table 1: Comparison of VERITAS with existing web-search benchmarks. "Wide degree" indicates breadth of search space (stars). VERITAS uniquely combines extreme breadth, exact automated verification, and no human dependence, whereas prior benchmarks trade off coverage against verification difficulty.

of high-entropy queries. First, relevant information is **severely fragmented** across many loosely connected web sources, with no single canonical page from which to harvest complete answer sets. Second, such tasks provide **no principled stopping rule**: neither annotators nor agents can know when further search will yield additional items. This combination makes literal completeness unattainable at web scale, so any human-constructed ground truth is almost guaranteed to be incomplete—and evaluation protocols may inadvertently penalize models that exceed human coverage while rewarding those that simply reproduce human omissions.

Consider the query: "Has the virtual singer KAF ever hosted a radio show, and if so, which show has the most episodes?" In a pilot analysis, one annotator concluded KAF has never hosted any show; another identified a short series with few episodes; a more systematic audit later found multiple series with substantially more episodes. This discrepancy reflects not human error, but the inherent difficulty—annotators face the same fragmentation and stopping challenges that define the task itself.

Addressing this paradox requires reformulating the evaluation problem, rather than attempting to obtain ever more exhaustive annotations.

3.2 Computational Equivalence

The ill-posedness of "list all X" benchmarks stems from annotation limitations, not evaluation goals; any task that forces the same high-entropy search without exploitable shortcuts is computationally equivalent.

From an evaluation perspective, the intent is not literally to obtain exhaustive lists, but to probe two deeper capabilities. First, an effective agent should carry out **wide and systematic search** over a large, loosely structured space, so that, given sufficient budget, the probability of missing salient items is small. Second, it should maintain **calibrated residual uncertainty**, distinguishing between "the space is probably covered" and "further exploration is still warranted" when deciding whether to stop. In this view, "list all X" is merely one syntactic vehicle for these desiderata. Our goal is to decouple the underlying search-and-coverage challenge from the brittle requirement of exact enumeration.

A simple analogy clarifies this equivalence: searching for the Ace of Spades in a well-shuffled deck. Whether we ask to "list all 52 cards" or to "find the Ace of Spades," the worst-case search cost is the same—one must inspect cards sequentially until the target appears, requiring $\Theta(N)$ operations over an unstructured space of size N. The identity of the target provides no algorithmic shortcut; the difficulty arises from traversing an unstructured space.

We apply the same idea to high-entropy web queries. Instead of "List all songs by artist X," we ask "Find the unique song by artist X whose title satisfies property Y." When property Y is **non-optimizable**—offering no usable signal to retrieval mechanisms—the agent must approximate the original $O(N)$ exhaustive search: gather candidates and check each against Y. By choosing Y to be *verifiable but unoptimizable*, we force agents to engage with the same high-entropy search problem while allowing evaluation to depend on a single,

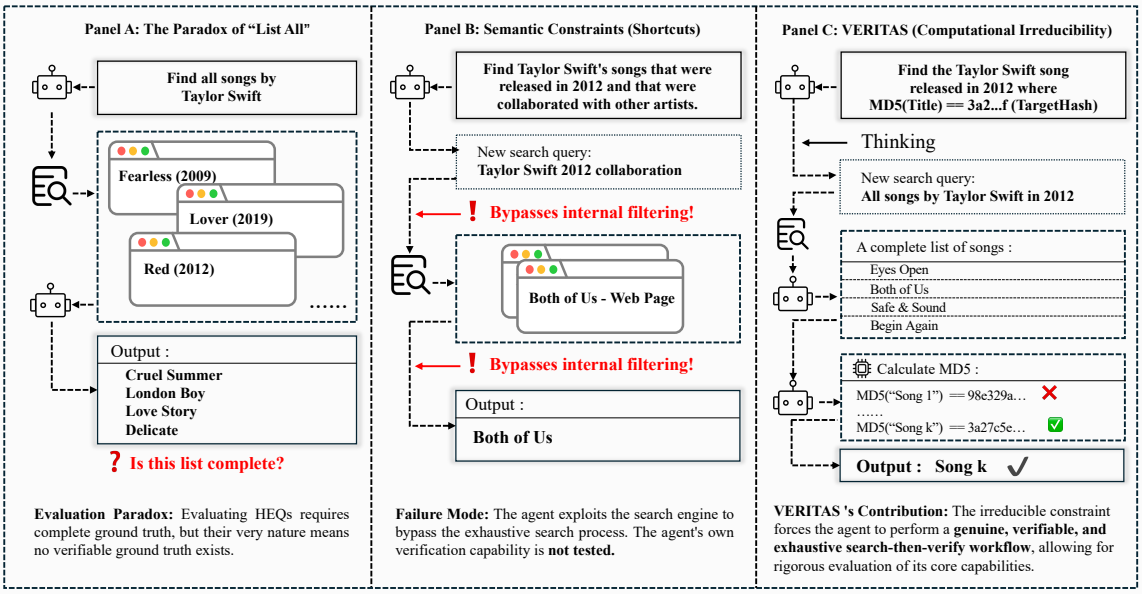

Figure 1: Evaluation paradox for high-entropy queries and our reformulation. (a) Exhaustive annotation is intractable. (b) Naively strengthening semantic constraints does not resolve the coverage problem. (c) Reformulating the task around automatically verifiable, non-optimizable constraints preserves exhaustive-search difficulty while enabling exact verification.

easily checked answer rather than a fragile exhaustive list.

Concretely, our benchmark instantiates Y using automatically verifiable yet non-optimizable constraints on item identifiers or derived attributes, so that success reduces to finding a small number of constraint-satisfying items in a large unstructured space. Figure 1 provides a high-level schematic of this shift from ill-posed exhaustive annotation to a constrained, verifiable search task. The next subsection formalizes this abstract model and analyzes its scaling behavior.

3.3 Asymptotic Analysis

We now formalize why finding k sparse constraint-satisfying items under a non-optimizable condition is computationally comparable to full enumeration. This provides a tractable model for interpreting our benchmark results.

Problem setup: Given N items in an unstructured search space, $k \leq N$ items satisfy a non-optimizable constraint. An agent with search budget T must find **all** k matching items. Success requires k correct items; partial matches count as failure.

Expected search cost: Assume the k target items are uniformly distributed among N positions. Under systematic or random search without replacement, the expected position of the last (k-th) discovered target is:

$$\mathbb{E}[L_k] = \frac{(N+1)\cdot k}{k+1}. \tag{1}$$

This is the standard order-statistics result: if the k targets occupy k random positions among N items, the expected position of the last one is $(N+1)\cdot k/(k+1)$.

Asymptotic behavior: For any fixed $k \geq 1$, $\mathbb{E}[L_k] = \Theta(N)$. Moreover,

$$\text{Relative cost: } \frac{\mathbb{E}[L_k]}{N} = \frac{k}{k+1} \xrightarrow{k\to\infty} 1. \tag{2}$$

Even for small k, the expected search cost is substantial: $\mathbb{E}[L_1] = (N+1)/2 \approx N/2$, $\mathbb{E}[L_2] = 2(N+1)/3 \approx 2N/3$, $\mathbb{E}[L_4] = 4(N+1)/5 \approx 4N/5$. Finding even a single sparse item therefore requires searching about half the space on average.

Success probability under budget constraints: Given finite search budget $T < N$, the probability of finding all k targets follows a hypergeometric distribution. If the agent searches exactly T items uniformly at random,

$$P_{\text{success}}(k,T,N) = \frac{\binom{N-k}{T-k}}{\binom{N}{T}} = \prod_{j=0}^{k-1} \frac{T-j}{N-j}. \tag{3}$$

For large N with small k and $T \ll N$, this probability can be approximated by

$$P_{\text{success}}(k,T,N) \approx \left(\frac{T}{N}\right)^k. \tag{4}$$

This expression highlights a sharp dependence on k: as k increases, success probability decreases roughly exponentially unless T is on the order of N. Specifically:

- $k = 1$: $P \approx T/N$ (linear dependence)
- $k = 2$: $P \approx (T/N)^2$ (quadratic drop)
- $k = 4$: $P \approx (T/N)^4$ (quartic drop)
- $k = N$: $P = 1$ iff $T = N$ (deterministic threshold).

Controlling difficulty via search space size: The success probability formula $P_{\text{success}}(k, T, N) \approx (T/N)^k$ reveals two independent mechanisms for controlling task difficulty: adjusting k (number of targets) or N (search space size). While the previous analysis focused on varying k, our benchmark design also exploits N as a difficulty lever.

Consider our Medium-tier tasks, which require searching across 7 independent sources (e.g., finding which of 7 artists has a song satisfying the target constraint). If each artist has approximately n songs, the effective search space becomes $N = 7n$, substantially larger than a single-artist search with $N = n$. For fixed $k = 1$ and search budget T, the success probability scales as:

$$\frac{P_{\text{single}}}{P_{\text{medium}}} = \frac{T/n}{T/(7n)} = 7. \tag{5}$$

This $7\times$ difficulty multiplier arises purely from increasing N, demonstrating that enlarging the search space (the denominator in $(T/N)^k$) is as effective as increasing the numerator k for creating challenging tasks. The Hard and Extra Hard tiers further increase N through deep attribute extraction and nested task hierarchies, respectively, creating progressively larger search spaces while maintaining irreducibility under the chosen non-optimizable constraints.

Taken together, these observations show that our transformation—from "enumerate all" to "find k constraint-satisfying items"—preserves the $O(N)$ computational irreducibility of the task while enabling deterministic verification. The non-optimizable constraint removes optimization shortcuts without altering the fundamental search complexity. Moreover, the $(T/N)^k$ framework provides principled control over difficulty through both k and N, enabling fine-grained benchmark design.

We now turn to how to construct constraints that realize this irreducible search behavior in practice.

3.4 Semantic Constraints

Semantic constraints fail as non-optimizable barriers: modern systems can delegate them to search engines, collapsing the intended $O(N)$ exhaustive search into $O(1)$ lookup.

The most direct response to the evaluation paradox is to introduce additional semantic constraints on target items. For instance, we might ask: "Find a Jay Chou song released in 2003, with a ballad style, lasting over four minutes, and featuring piano." At face value, this appears to require enumerating candidates and verifying each. In practice, however, indexing and filtering mechanisms explicitly exploit semantic information, so temporal, categorical, or relational constraints become **signals for optimization** rather than obstacles. The model can submit the entire constraint set as a query and retrieve a small candidate pool directly.

More broadly, many "fuzzy" web tasks decompose into two stages: first *enumerate* a broad candidate space, then *filter* using semantic conditions. The intrinsic difficulty lies in enumeration; once candidates are available, filtering is straightforward. Our benchmark therefore targets enumeration directly. To recover computational equivalence with high-entropy queries while avoiding semantic shortcuts, we require constraints that are **verifiable but unoptimizable**: easily checked once a candidate is known, yet providing no useful guidance to search engines.

3.5 Cryptographic Hashes

Cryptographic hash functions provide ideal non-optimizable constraints through two properties: one-way verification and semantic nullity.

One-way verification: Pre-image resistance makes reversing the hash computationally infeasible, yet verification is trivial—hash the candidate and compare. This is a lock that only opens when you already have the key.

Semantic nullity: Hash outputs bear no exploitable relationship to the semantic content of their inputs. The avalanche effect ensures that small input changes produce seemingly unrelated outputs, rendering the hash useless as a search query. Search engines cannot leverage it to narrow the search space.

Under these conditions, solving the task requires approximate $O(N)$ exhaustive search. Models cannot use the hash value to guide retrieval; they must enumerate candidate items and verify each one.

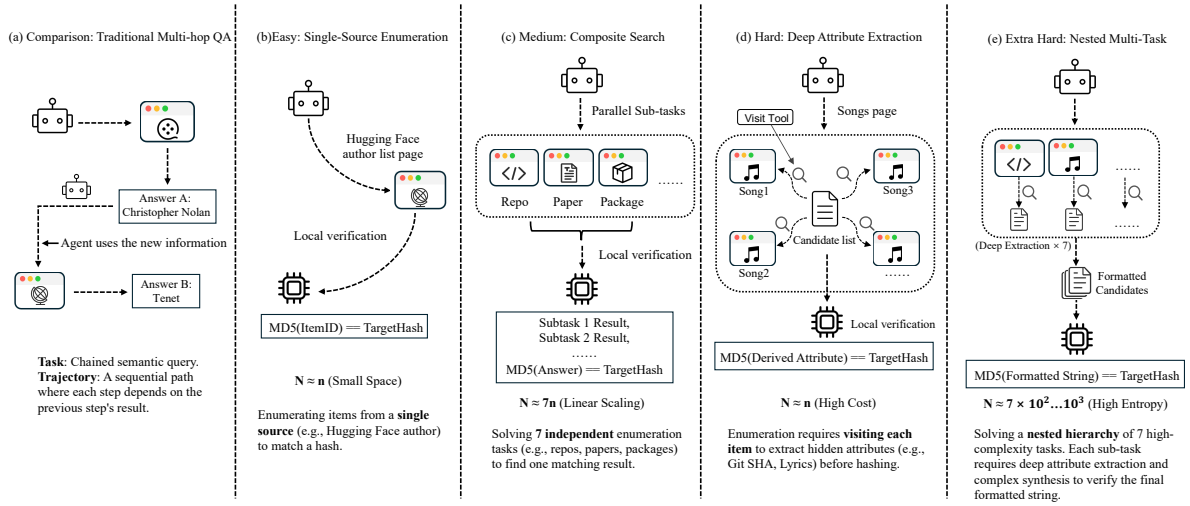

Figure 2: Progressive difficulty tiers in VERITAS. Higher tiers enlarge the search space and computational burden while preserving hash-based irreducibility.

This transformation preserves the underlying computational challenge while enabling reliable, scalable verification. A hash-constrained task with N implicit candidates and k true pre-images can be modeled as selecting k targets uniformly at random from an unstructured space of size N, matching the abstract setting analyzed previously.

3.6 VERITAS Tasks

VERITAS implements four progressive difficulty tiers that systematically stress-test exhaustive search under increasing computational irreducibility (Figure 2).

Each tier enlarges the effective search space N or increases task complexity while preserving hash-based irreducibility. This provides principled difficulty control: per the $(T/N)^k$ framework, both expanding N and increasing structural complexity reduce success probability for budget-constrained agents.

Easy (Single Source): Enumerate all items from a single source (e.g., "all models by author `google` on HuggingFace") and identify the item whose identifier's MD5 hash matches the target. Assesses basic enumeration discipline.

Medium (Composite Search): Seven independent enumeration subtasks; the agent must solve each, then determine which result produces the target MD5 hash. Evaluates coordination of multi-stage exhaustive search.

Hard (Deep Attribute Extraction): Enumerate items from a source and identify which item's **derived attribute** (e.g., a Git commit SHA) yields the target hash. Emphasizes deep attribute extraction from distributed sources.

Extra Hard (Nested Multi-Task): Seven high-complexity enumeration tasks; construct complete formatted answers (e.g., "RepoName $\rightarrow$ SHA") and determine which formatted string has the target hash. Probes sustained search across deeply nested task hierarchies.

4 Experiments

VERITAS reveals a clear capability hierarchy: frontier models (GPT-5, Gemini-3-Pro) substantially outperform others, yet even they struggle on Extra Hard, with success rates dropping by 2–5×. We evaluate nine models across Medium, Hard, and Extra Hard tiers to characterize both capability gaps and failure modes under computationally irreducible constraints.

4.1 Agent Tools

Agents are equipped with four tools—Search, Visit, Exec Python, and Answer—providing capabilities comparable to human web researchers.

Search: Provides access to Google search, returning top-10 results (title, snippet, URL) for each query. Supports multiple simultaneous queries.

Visit: Accesses specific web pages with dedicated extraction goals. Following WebSailor's approach, we use Qwen3-235B-A22 to extract goal-relevant information from each page.

Exec Python: Executes Python code for computations, including MD5 hash verification of candi-

Table 2: Performance on VERITAS benchmark across difficulty tiers. We report Pass@4 success rate (%) and the average number of tool calls per attempt, split into Search and Visit. Best results are highlighted in **bold**.

Model	Medium				Hard				Ex-Hard			
	Success		Tool Calls		Success		Tool Calls		Success		Tool Calls	
	Pass@4	Avg@4	Search	Visit	Pass@4	Avg@4	Search	Visit	Pass@4	Avg@4	Search	Visit
Open-Source Models												
GLM-4	**17.9**	5.7	23.4	10.1	12.2	5.6	14.9	8.8	3.3	2.5	23.2	12.2
MiniMax-M2	12.5	4.4	14.9	8.7	6.1	1.5	17.4	7.8	3.4	0.8	17.7	10.0
DeepSeek-v3.1	10.3	3.8	15.5	8.8	**14.3**	7.1	9.2	4.1	3.4	0.9	16.0	9.2
Qwen3-235B-A22	5.3	2.6	16.8	13.1	6.2	2.6	8.0	3.9	**3.6**	2.6	16.9	12.4
Kimi-K2	2.5	0.6	16.2	8.5	10.2	2.6	18.0	6.5	0.0	0.0	26.9	9.5
Closed-Source Models												
GPT-4o	6.0	1.5	5.3	4.9	2.4	2.4	2.2	2.1	0.0	0.0	5.1	4.7
Gemini-2.5-Pro	12.5	6.2	14.8	9.9	8.2	3.6	9.6	4.4	3.3	1.7	14.4	10.8
Gemini-3-Pro	30.0	18.8	84.7	6.8	**36.7**	21.9	92.5	6.8	6.7	3.3	125.8	9.3
GPT-5	**47.5**	27.5	7.8	101.8	32.7	25.5	7.3	16.4	**20.0**	20.0	9.3	27.8

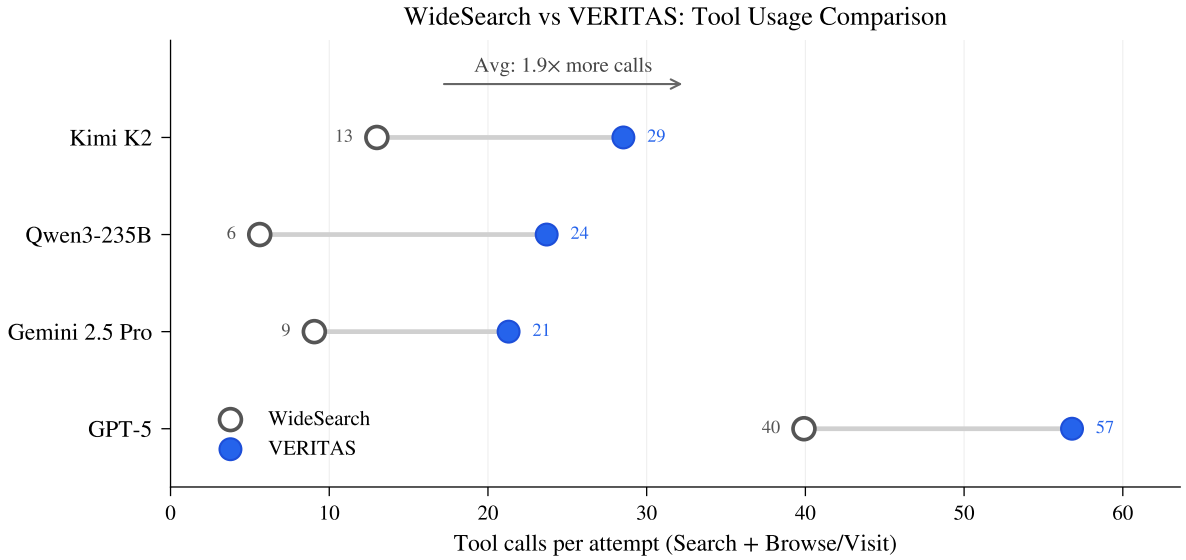

Figure 3: Tool calls per attempt for models evaluated on both WideSearch and VERITAS. VERITAS consistently elicits 1.4×–4.2× more tool calls than WideSearch, yet success rates remain low (Table 2), confirming that VERITAS removes semantic shortcuts and forces genuine exhaustive enumeration.

date items.

Answer: Submits the final solution for evaluation.

4.2 Experimental Setup

We evaluate across Medium/Hard/Extra Hard tiers on balanced multilingual queries, with strict all-or-nothing success criteria.

We deliberately exclude Easy tier to focus on realistic high-entropy challenges. The dataset maintains balanced distribution across English, Chinese, and Japanese, ensuring robust cross-lingual evaluation.

We evaluate nine models spanning diverse architectures and capability levels: **GLM-4**, **MiniMax-M2**, **DeepSeek-v3.1**, **Qwen3-235B-A22**, **Kimi-K2**, **GPT-4o**, **Gemini-2.5-Pro**, and the latest frontier models **GPT-5** and **Gemini-3-Pro**. Success requires finding *all* items matching the target MD5 hash—partial matches count as failure.

We report **Pass@4** (success rate across 4 attempts) and track tool usage via average **Search** and **Visit** calls per attempt.

4.3 Main Results

VERITAS effectively discriminates model capabilities: frontier models (GPT-5, Gemini-3-Pro) substantially outperform others, yet even they see dramatic performance collapse on Extra Hard.

Table 2 reveals a clear **capability hierarchy**. GPT-5 achieves 47.5% Pass@4 on Medium and 32.7% on Hard—far surpassing other models. Gemini-3-Pro follows with 30.0% and 36.7% respectively. In contrast, the remaining seven models cluster between 2.5–17.9% on Medium and 2.4–14.3% on Hard. This stark separation demonstrates that VERITAS provides meaningful signal for distinguishing frontier capabilities from current-generation models.

Crucially, **Extra Hard remains challenging even for frontier models**. GPT-5 drops from 47.5% (Medium) to 20.0% (Extra Hard)—a $2.4\times$ reduction. Gemini-3-Pro drops more sharply from 36.7% (Hard) to just 6.7% (Extra Hard). Meanwhile, most other models collapse to 0–3.6% on Extra Hard, with GPT-4o and Kimi-K2 achieving exactly 0%. This pattern confirms that Extra Hard tier probes a genuine capability frontier: it separates GPT-5 (the only model maintaining double-digit performance) from all others, while still exposing substantial room for improvement.

The **difficulty gradient works as designed**: success rates drop consistently from Medium to Hard to Extra Hard across all models. This validates our $(T/N)^k$ theoretical framework—as the effective search space N expands through composite search, deep attribute extraction, and nested hierarchies, success probability decreases predictably.

Finally, the gap between Pass@4 and Avg@4 reveals search strategy quality. GPT-5 shows notably higher consistency (Avg@4 of 27.5%, 25.5%, 20.0% across tiers) compared to other models where Avg@4 often falls far below Pass@4, indicating that weaker models occasionally stumble upon solutions but lack systematic search strategies.

4.4 Scaling under Hash Constraints

Models invest substantial search effort (20–30+ tool calls per query) yet achieve $<$12% success, confirming that VERITAS forces genuine exhaustive enumeration without semantic shortcuts.

Tool-usage statistics reveal a striking effort-outcome mismatch. GLM-4 issues on average 31.3 tool calls per query (19.8 Search, 10.0 Visit), Kimi-K2 28.5, and DeepSeek-v3.1 21.9—far exceeding typical single-answer QA benchmarks such as GAIA (average $<$10 calls). Despite examining dozens of web pages, success rates remain below 12%. Models recognize the need for exhaustive exploration but fail to achieve sufficient coverage.

Figure 3 provides direct evidence: for the same underlying models, VERITAS elicits $1.4\times$–$4.2\times$ more tool calls than WideSearch, yet Pass@4 remains low. This gap underscores that VERITAS removes semantic shortcuts and forces agents into genuine enumeration regimes.

This pattern precisely matches the $(T/N)^k$ scaling analysis: as effective search space N grows from Medium to Extra Hard while tool budgets remain fixed, success probability collapses across all models. VERITAS thus serves as a sharper stress test of exhaustive-search capability rather than surface-level browsing.

5 Conclusion

We argued that the core difficulty in modern web search and "deep research" is not the length of reasoning chains, but the *entropy* of the search space: the need to enumerate large, weakly structured candidate sets and to know when it is safe to stop. Through theoretical analysis, we showed that finding a few sparse, hash-constrained targets is computationally equivalent to exhaustive enumeration, providing a path to automatically verifiable evaluation of this capability.

Building on this analysis, VERITAS instantiates cryptographic hash constraints over realistic, web-scale high-entropy queries to create tasks where any successful agent must effectively "list everything that could possibly match" before applying a cheap verification step. Experiments across strong web agents reveal that, despite heavy tool use, models still collapse on these tasks as search spaces grow, highlighting a fundamental gap between current systems and truly exhaustive search.

Looking forward, the same properties that make VERITAS attractive as a benchmark—infinite automatic generation, exact labels, and tunable difficulty—also make it a promising source of training data for future search agents, for example via reinforcement learning or RL-style post-training. We view our work as a first step towards systematically teaching models not just to find good answers,

but to approach the harder goal of finding *all* relevant answers under uncertainty.

Limitations

VERITAS currently serves purely as an evaluation benchmark. Although its automatic, verifiable structure makes it well suited for reinforcement learning or other post-training methods, we have not yet applied such training to models using VERITAS-generated data; exploring this direction is an important target for future work.

Acknowledgements

This work was by National Key Program of National Natural Science of China (Grant No.82430108).

References

Doubao. 2025. Doubao deep research.

Gemini. 2025. Gemini deep research.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. 2024. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Kimi. 2025. Kimi deep research.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Tian Lan, Bin Zhu, Qianghuai Jia, Junyang Ren, Haijun Li, Longyue Wang, Zhao Xu, Weihua Luo, and Kaifu Zhang. 2025. Deepwidesearch: Benchmarking depth and width in agentic information seeking. *arXiv preprint arXiv:2510.20168*.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, Weizhou Shen, Junkai Zhang, Dingchu Zhang, Xixi Wu, Yong Jiang, Ming Yan, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025a. Websailor: Navigating superhuman reasoning for web agent. *arXiv preprint arXiv:2507.02592*.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. 2025b. Webthinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776*.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, and 1 others. 2023. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*.

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*.

OpenAI. 2025. Deep research system card.

Perplexity. 2025. Perplexity deep research.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*.

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, and 1 others. Webshaper: Agentically data synthesizing via information-seeking formalization, 2025. *URL https://arxiv. org/abs/2507.15061*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

J Wei, Z Sun, S Papay, S McKinney, J Han, I Fulford, HW Chung, AT Passos, W Fedus, and A Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents, 2025a. *URL https://arxiv. org/abs/2504.12516*.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*.

Ryan Wong, Jiawei Wang, Junjie Zhao, Li Chen, Yan Gao, Long Zhang, Xuan Zhou, Zuo Wang, Kai Xiang, Ge Zhang, and 1 others. 2025. Widesearch: Benchmarking agentic broad info-seeking. *arXiv preprint arXiv:2508.07999*.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, and 1 others. 2025. Webdancer: Towards autonomous information seeking agency. *arXiv preprint arXiv:2505.22648*.

x.ai. 2025. Grok 3 beta — the age of reasoning agents.

Yunjia Xi, Jianghao Lin, Yongzhao Xiao, Zheli Zhou, Rong Shan, Te Gao, Jiachen Zhu, Weiwen Liu, Yong Yu, and Weinan Zhang. 2025. A survey of llm-based deep search agents: Paradigm, optimization, evaluation, and challenges. *arXiv preprint arXiv:2508.05668*.

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*.

Weizhi Zhang, Yangning Li, Yuanchen Bei, Junyu Luo, Guancheng Wan, Liangwei Yang, Chenxuan Xie, Yuyao Yang, Wei-Chieh Huang, Chunyu Miao, Henry Peng Zou, Xiao Luo, Yusheng Zhao, Yankai Chen, Chunkit Chan, Peilin Zhou, Xinyang Zhang, Chenwei Zhang, Jingbo Shang, and 4 others. 2025. From web search towards agentic deep research: Incentivizing search with reasoning agents. *arXiv preprint arXiv:2506.18959*.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, and 1 others. 2025. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese. *arXiv preprint arXiv:2504.19314*.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, and 1 others. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.