The Web Tool Trap: Understanding and Mitigating Over-Reliance in Browsing Agents

Anonymous ACL submission

Abstract

Large Language Model (LLM) agents that can gather knowledge by browsing the web are becoming increasingly useful and important. However, their effectiveness is often hindered by an imperfect integration of internal knowledge and external tools. We introduce BrowseBench and present the first systematic investigation into the over-reliance patterns of browsing agents on web tools. Through controlled experiments, we identify three distinct failure modes: (1) Excessive Conservatism, where agents unnecessarily invoke search tools for information already mastered in their training; (2) Over-trust in Web Sources, where agents apply inconsistent standards by questioning reliable internal knowledge while uncritically accepting web-retrieved content; and (3) Planning Deficiency, characterized by a lack of search planning and decomposition strategies for complex queries. These contradictions result in inefficient information processing, resource waste, and erroneous conclusions. To address these challenges, we propose three mitigation strategies: Direct Preference Optimization (DPO) to calibrate search decision boundaries, Attention Refinement to filter retrieved content, and Hierarchical Query Decomposition to improve multi-round tool coordination. Experiments demonstrate that our interventions significantly reduce over-reliance behaviors and enhance performance. Our work provides critical insights for the deployment of robust, tool-augmented LLMs in real-world applications.

1 Introduction

Large Language Models (LLMs) have evolved dramatically from simple conversational chatbots to sophisticated Artificial Intelligence (AI) agents capable of interacting with external tools and environments (Xi et al., 2025; Li, 2025; Yao et al., 2023). Early LLMs (Brown et al., 2020; Bai et al., 2023; Touvron et al., 2023) primarily functioned as text generators, responding to queries based solely on their parametric knowledge acquired during training (Guo et al., 2025b). However, the integration of tool-use capabilities has fundamentally transformed these models into autonomous agents that can access real-time information, execute computations, and interact with various Application Programming Interfaces (APIs) (Lewis et al., 2020; Li et al., 2023, 2025; Ren et al., 2025a,b). Among these tools, web search has emerged as particularly prominent, enabling LLMs to overcome knowledge cutoff limitations and access up-to-date information without re-training and fine-tuning. This evolution has led to systems like GPT-4 (Achiam et al., 2023) with browsing capabilities, Claude[1] with web search integration, and various open-source frameworks that augment LLMs with tool-calling abilities, marking a significant paradigm shift in how AI systems retrieve and process information.

The phenomenon of tool dependency in AI systems bears striking parallels to human cognitive behavior. Risko and Gilbert (2016) systematically articulated the theory of cognitive offloading in their seminal review published in Trends in Cognitive Sciences, describing how humans strategically transfer cognitive burdens to external tools and resources to optimize mental processing. This theoretical framework provides a compelling lens through which to examine AI behavior.

However, current evaluation benchmarks for tool-augmented LLMs exhibit critical limitations in addressing the phenomenon of tool over-reliance. Existing frameworks such as ToolBench (Qin et al., 2024a), WebGPT evaluations (Nakano et al., 2022), API-Bank (Chen et al., 2025), and τ-bench (Yao et al., 2025) predominantly focus on correctness metrics – whether the model successfully uses tools to arrive at accurate answers. While correctness is undoubtedly important, these benchmarks fail

[1] https://www.anthropic.com/news/claude-3-7-sonnet

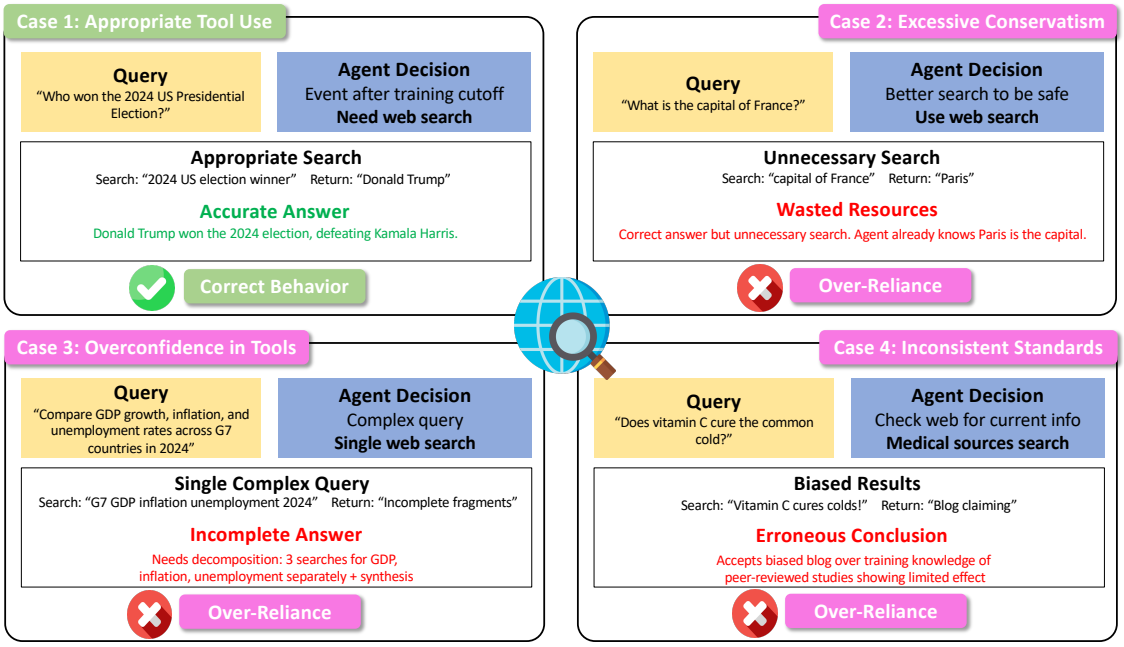

Figure 1: Overview of Browsing Agent Behaviors: Normal Tool Usage vs. Three Over-reliance Failure Modes.

to capture the nuanced decision-making required for optimal tool usage. They neglect to evaluate whether tool invocation was necessary in the first place, ignore the critical balance between leveraging internal knowledge and external resources, and lack metrics for assessing the efficiency and appropriateness of tool-use patterns. This myopic focus on accuracy overlooks the fundamental question of when tools should be used versus when the model's parametric knowledge suffices, rendering them inadequate for assessing web search tool over-reliance.

Beyond evaluation gaps where current benchmarks neglect the critical question of when tools should be used versus when parametric knowledge suffices, existing mitigation strategies also present limitations. While previous research has separately addressed the over-reliance on call process (Xu et al., 2025a) and the over-reliance on call output (Fang et al., 2024), no research has proposed a unified strategy to mitigate both facets of web tool over-reliance simultaneously. Therefore, significant research gaps remain in developing effective unified mitigation strategies, constructing appropriate evaluation datasets, and improving the explainability of web tool over-reliance in LLMs.

To address those research gaps, we propose the first systematic investigation into the over-reliance patterns of web tools. We create BrowseBench, a dataset as a foundation to rigorously evaluate how browsing agents interact with web search tools when faced with a diverse range of questions. We ground our analysis in 1,500 realistic information-seeking scenarios to identify and understand the failure modes associated with tool over-reliance.

As shown in Figure 1, our study reveals three reasons for the failure of web tools. First, models exhibit conservative behavior by performing unnecessary searches due to knowledge boundary ambiguity, contextual misdirection from temporal markers, and defensive cognitive patterns triggered by epistemic uncertainty. Second, models overtrust web sources by uncritically accepting retrieved content, with attention mechanisms prioritizing linguistic coherence and structural formatting over factual accuracy. Third, models lack strategic planning capabilities, either compressing multi-faceted queries into semantically overloaded single searches or generating redundant parallel searches without systematic decomposition of sequential dependencies.

In response to these failure modes, we develop three mitigation strategies. First, Direct Preference Optimization (DPO) (Rafailov et al., 2023) enables models to learn nuanced boundaries for search decisions. Second, Attention Refinement(AR) helps agents focus on relevant information within retrieved content. Third, Hierarchical Query Decomposition(HQD) improves multi-round tool coordination and query decomposition. Our experiments demonstrate that these interventions significantly reduce over-reliance behaviors. The results reveal that while current browsing agents' over-reliance

on web tools can amplify errors, carefully designed training strategies can substantially improve their tool usage patterns. These findings have important implications for deploying tool-augmented LLMs in real-world applications.

As such, the contributions of this paper are:

1. We conduct the first systematic investigation into the over-reliance patterns of browsing agents on web tools.

2. We provide a comprehensive analysis of browsing agents, identifying and characterizing three key failure modes.

3. We propose and validate three effective mitigation techniques, which demonstrably improve the efficiency and reliability of tool-augmented LLMs.

2 Related Work

2.1 LLMs as Agents

The integration of Large Language Models (LLMs) with external tools has emerged as a pivotal advancement in enhancing their capabilities beyond text generation. ReAct (Yao et al., 2023) pioneered the synergistic combination of reasoning and acting, enabling LLMs to interleave thought processes with tool interactions. This paradigm shift has been further explored through frameworks like Toolformer (Schick et al., 2023), which teaches language models to autonomously decide when and how to use external tools through self-supervised learning. Recent advances have expanded the tool-use paradigm across diverse applications. WebGPT (Nakano et al., 2022) demonstrates web browsing capabilities, while systems like ChatGPT Plugins and GPT-4's function calling capabilities have brought tool use to production environments. Gorilla (Patil et al., 2023) specializes in API calls, achieving state-of-the-art performance in selecting and invoking appropriate APIs from large repositories. ToolLLM (Qin et al., 2024b) introduces a comprehensive framework for tool learning with over 16,000 real-world APIs, highlighting the growing complexity of tool-augmented systems. The architectural evolution of tool-using agents has progressed from simple retrieval augmentation to sophisticated multi-agent systems. AutoGPT [2] and BabyAGI [3] represent autonomous agents capable of decomposing complex tasks and orchestrating multiple tool interactions. However, this increased capability introduces new challenges, particularly in determining when tool use is genuinely necessary versus when the model's parametric knowledge suffices—a critical gap that existing literature has yet to systematically address.

[2] https://github.com/Significant-Gravitas/AutoGPT

[3] https://github.com/yoheinakajima/babyagi

2.2 Evaluation Benchmarks of Tool Use

The evaluation of LLMs' tool-use capabilities has evolved along two primary dimensions: closed-environment benchmarks with predefined tool sets and open-environment evaluations simulating real-world conditions. (i) Closed Environment Benchmarks. Berkeley Function Calling Leaderboard (BFCL) (Patil et al., 2025) comprehensively evaluates function calling accuracy across multiple programming languages. τ-bench (Yao et al., 2025) introduces temporal reasoning tasks requiring real-time information retrieval. AceBench (Chen et al., 2025) and ToolBench (Qin et al., 2024a) focus on complex tool composition and large-scale tool selection respectively. While these benchmarks rigorously test tool-use accuracy, they assume tool invocation is inherently desirable, measuring how well models use tools rather than whether they should. (ii) Open Environment Benchmarks. SearchArena (Miroyan et al., 2025) evaluates real-world search capabilities with authentic web scenarios. WebArena (Zhou et al., 2024) and BrowserGym (de Chezelles et al., 2025) provide realistic web environments for testing multi-step tasks. These benchmarks successfully capture real-world complexity but implicitly assume external information gathering is always necessary, overlooking cases where parametric knowledge suffices.

2.3 Tool Over-reliance of LLMs

LLMs often exhibit an over-reliance on tools and excessive self-confidence. Kokane et al. (2025) and Xu et al. (2025b) demonstrate that LLMs exhibit a tendency towards repeated API calls, even when the tool should not be invoked again in subsequent steps. Furthermore, Xu et al. (2025a) demonstrate that LLMs incline towards calling external tools even when addressing simple problems entirely solvable through their own parameterised knowledge, and propose an alignment framework consisting of knowledge boundary estimation and knowledge boundary modeling to make LLMs more efficient in tool calling. Despite these observations, the explainability of this over-reliance has not been

a primary focus of previous research.

3 BrowseBench

3.1 Dataset Construction Framework

As shown in Figure 2, we develop a linguistically grounded test dataset that leverages different parts of speech to create diverse and naturalistic queries to systematically evaluate tool over-reliance in LLMs. All prompts are detailed in Appendix A.

Generation: Starting with 150 seed queries collected from search-related benchmarks, we synthesize more data covering diverse domains and types. The collected seed queries undergo a sophisticated augmentation process through part-of-speech-based modifications guided by LLMs. Our approach instructs Qwen-2.5-72B-Instruct to analyze the grammatical structure of a seed query, strategically modifying its parts of speech to generate a diverse set of new queries.

Deduplication: Following data generation, we implement a deduplication process to ensure dataset diversity and eliminate redundant queries.

LLM Labeling: Subsequently, we employ Qwen-2.5-72B-Instruct to automatically label each query with relevant keywords and domain classifications, providing structured metadata for downstream analysis.

Verification: To ensure data quality, we conduct manual quality assurance where human annotators verify the logical consistency of queries and filter out those lacking definitive answers, maintaining the dataset's coherence and answerability.

Annotation: Finally, we recruit experienced crowd-sourcing annotators as domain experts to annotate queries, all of whom hold at least a bachelor's degree. Experts annotate validated queries while we record their decision paths for web tools usage and final answers. These expert demonstrations serve as references for subsequent analysis of model behavior.

For each model and each query, we validate that the model can answer the query by converting the expert's decision path into context for the model. The final dataset comprises 1,500 queries equally distributed across five domains (300 queries each): Culture & Society (CS), Science & Technology (ST), Biology & Medicine (BM), Environment (EN), and Finance (FI). To enable multi-dimensional analysis, each query is annotated with 3-6 keywords for fine-grained categorization, with a balanced distribution: 30% of queries have 3 keywords, 30% have 4 keywords, 20% have 5 keywords, and 20% have 6 keywords. Further details are illustrated in Appendix B.

3.2 Evaluation Metrics

Unnecessary Search Rate (USR). This metric measures the proportion of search queries that do not contribute meaningful information to the final response. It quantifies the agent's tendency to perform redundant or irrelevant searches when the information could be obtained from its internal knowledge or previous search results. The USR is calculated as:

$$\text{USR} = \frac{N_{\text{unnecessary}}}{N_{\text{total}}} \times 100\% \tag{1}$$

where N_{total} represents the total number of search queries performed, $N_{\text{unnecessary}}$ denotes the number of searches deemed unnecessary.

Interference Information Incorporation Rate (IIIR). This metric quantifies the proportion of retrieved interference information that the LLM uncritically adopts into its responses, directly measuring overtrust in noisy or misleading web sources. The IIIR is defined as:

$$\text{IIIR} = \frac{N_{\text{interference\_incorporated}}}{N_{\text{interference\_retrieved}}} \times 100\% \tag{2}$$

where $N_{\text{interference\_incorporated}}$ represents the number of interfering claims from retrieved sources that appear in the LLM's response (either verbatim or paraphrased), and $N_{\text{interference\_retrieved}}$ is the total number of interfering claims present in all retrieved sources (verified against ground truth). A high IIIR indicates a systematic failure in robustness, revealing the model's tendency to prioritize the linguistic coherence of retrieved interference over factual accuracy.

Task Decomposition Deviation (TDD). This metric evaluates the agent's ability to properly decompose complex queries into manageable subtasks by comparing its execution path against expert demonstrations. It captures whether the agent appropriately breaks down multi-faceted questions or attempts to solve them in overly simplified or complicated ways:

$$\begin{aligned} \text{TDD} &= \left| L_{\text{agent}} - L_{\text{expert}} \right| \\ &= \left| \sum_{i=1}^{T} \mathbb{I}[\text{step}_i] - L_{\text{expert}} \right| \end{aligned} \tag{3}$$

where L_{agent} is the number of decision steps (search or fetch operations) taken by the agent,

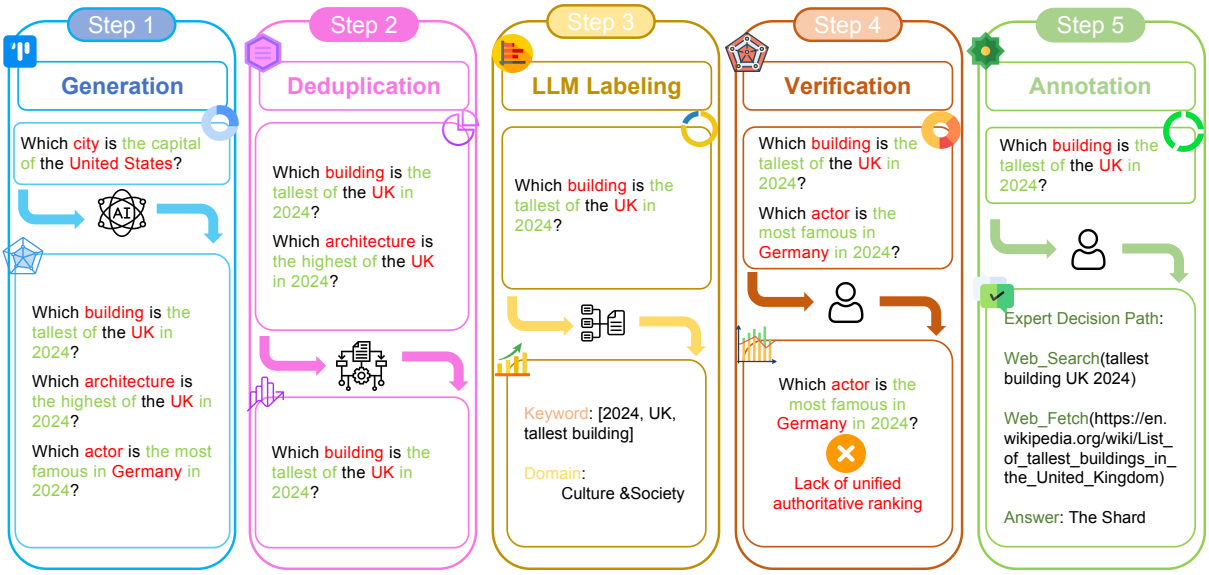

Figure 2: **Overview of Data Construction Framework. Generation:** Use Qwen-2.5-72B-Instruct to perform part-of-speech-based augmentation on 150 seed queries, strategically modifying grammatical elements to create diverse queries across domains; **Deduplication:** Eliminate redundant queries to ensure dataset diversity; **LLM Labeling:** Automatically annotate queries with keywords and domain classifications using Qwen-2.5-72B-Instruct; **Verification:** Human annotators verify logical consistency and filter out unanswerable queries; **Annotation:** Domain experts record web tool usage decisions and answers as reference demonstrations.The tutorial is provided in Appendix G.

L_{expert} represents the average number of steps taken by human experts for the same task, and T is the total number of steps in the agent's execution trace. A TDD close to 0 indicates expert-like task decomposition, while negative values suggest under-decomposition and positive values indicate over-decomposition.

4 Experiment

We consider both closed- and open-source models. Our evaluation is conducted under a greedy decoding and zero-shot setting to assess the capability of models to generate accurate answers without fine-tuning or few-shot demonstrations on our benchmark. For all models, we use the same minimalist prompt for comparison fairness. All experiments are conducted with NVIDIA H100 GPUs. All evaluation results are judged using GPT-4o-mini. All prompts are detailed in Appendix A.

We have equipped the model with web search capabilities through the Brave Search API [4] to simulate realistic information retrieval patterns. Brave Search is selected for its privacy-preserving architecture and security features, ensuring that search queries remain untracked and results are delivered without personalization bias. Each search query returns up to 20 results, providing the model with diverse information sources while maintaining computational efficiency. Furthermore, we have also implemented a complementary web_fetch function that enables the model to fetch raw HTML content from specific URLs. This dual-functionality approach allows the model to first discover relevant sources through search queries, then extract detailed information from selected webpages, mimicking the natural information-seeking patterns observed in human web browsing behavior. The calculation process for metrics can be found in Appendix C.

4.1 LLM is a Conservative

Case studies in Appendix D.1 identify three mechanisms triggering unnecessary web searches (USR), with significant domain variation shown in Table 1.

Knowledge Boundary Ambiguity. Models often fail to distinguish stable theoretical foundations from evolving empirical data. Specialized domains like BM (41.9%), ST (39.5%), and FI (39.6%) show significantly higher USR than CS (32.9%). Case 1 exemplifies this: Gemini-2.5-Pro searched for "electron transport chain steps" despite its status as fundamental biochemistry, leading to a BM USR 47.8% higher than its CS performance.

Contextual Misdirection. Temporal markers

[4] https://brave.com/search/api

Table 1: Performance metrics across all domains. CS: Culture & Society, ST: Science & Technology, BM: Biology & Medicine, EN: Environment, FI: Finance, Avg.: Overall Average. Our evaluation is conducted under a greedy decoding and zero-shot setting. For all models, we use the same minimalist prompt for comparison fairness.

Model	CS			ST			BM		
	USR	IIIR	TDD	USR	IIIR	TDD	USR	IIIR	TDD
Claude-3.7-Sonnet (Anthropic, 2025)	18.9	24.5	1.8	27.2	28.3	2.6	28.5	31.2	2.8
Grok-4 (xAI, 2025)	22.4	29.1	2.2	31.5	34.6	3.1	33.2	38.9	3.4
Gemini-2.5-Pro (Comanici et al., 2025)	22.8	29.8	2.3	32.1	35.2	3.2	33.7	39.6	3.5
Gemini-2.5-Flash (Comanici et al., 2025)	23.7	30.9	2.3	33.0	36.4	3.3	34.6	40.8	3.5
GPT-4o (OpenAI, 2024b)	25.3	32.7	2.4	35.8	38.5	3.3	37.2	42.1	3.6
DeepSeek-R1 (Guo et al., 2025a)	29.5	35.4	2.2	33.7	37.9	3.0	44.3	48.2	3.2
Kimi-K2 (Team et al., 2025)	26.5	38.6	2.7	37.3	42.8	3.8	38.6	46.5	4.0
GPT-o4-mini (OpenAI, 2024a)	28.9	41.2	2.5	39.6	45.5	3.3	41.0	49.2	3.7
Qwen2.5-72B-Instruct (Team, 2024)	38.2	52.3	3.4	50.5	58.7	4.5	52.8	63.4	4.7
Llama3.1-70B-Instruct (Dubey et al., 2024)	43.7	61.8	3.8	56.2	68.2	5.0	58.5	72.5	5.3
Model	EN			FI			Avg.		
	USR	IIIR	TDD	USR	IIIR	TDD	USR	IIIR	TDD
Claude-3.7-Sonnet	22.3	26.7	2.2	27.1	29.4	2.6	24.8	28.0	2.4
Grok-4	26.6	32.5	2.7	30.3	36.0	3.1	28.8	34.2	2.9
Gemini-2.5-Pro	27.0	33.2	2.8	30.9	36.8	3.2	29.3	34.9	3.0
Gemini-2.5-Flash	27.9	34.3	2.6	31.9	38.1	3.0	30.2	36.1	2.9
GPT-4o	29.6	35.9	2.9	34.6	40.3	3.3	32.5	37.9	3.1
DeepSeek-R1	34.7	41.5	2.6	42.5	47.6	3.0	37.0	42.1	2.8
Kimi-K2	31.2	39.7	3.3	37.0	44.2	3.7	34.1	42.4	3.5
GPT-o4-mini	33.4	42.3	3.1	39.4	46.9	3.6	36.5	45.0	3.2
Qwen2.5-72B-Instruct	44.7	56.8	4.0	50.3	62.1	4.4	47.3	58.7	4.2
Llama3.1-70B-Instruct	50.1	65.4	4.5	56.0	70.8	4.9	52.9	67.7	4.7

trigger retrieval even for axiomatic facts. In Case 2, Kimi-K2 searched for "CNN architecture before 2022," a well-documented historical topic. Such linguistic triggers contributed to Kimi-K2's ST USR of 37.3%, 40.8% higher than its CS baseline.

Cognitive Triggering Patterns. Epistemic qualifiers (e.g., "according to current research") prompt defensive searches regardless of knowledge stability. Open-source models exhibit particularly high USR; Llama3.1-70B-Instruct reached 52.9% overall, 113.3% worse than Claude-3.7-Sonnet. Even top-tier models show vulnerability, with Claude's BM USR being 50.8% higher than its CS rate. This systematic hypercaution in complex domains compromises efficiency without improving accuracy.

4.2 LLM Overtrusts Web Sources

LLMs exhibit a systematic tendency to adopt web-retrieved information without rigorous fact-checking. Our results in Table 1 quantify this via IIIR, measuring how frequently models uncritically propagate incorrect search results.

Widespread Uncritical Acceptance. IIIR reveals pervasive overtrust across all models (28.0%–67.7%). Open-source models are most vulnerable: Llama3.1-70B-Instruct reaches 67.7% IIIR, incorporating plausible-but-false information in two-thirds of cases, followed by Qwen2.5-72B-Instruct (58.7%). Proprietary models perform better but remain problematic: GPT-4o (37.9%), Gemini-2.5-Pro (34.9%), and Claude-3.7-Sonnet (28.0%). Despite a 2.4× performance gap, even the strongest model is misled over a quarter of the time by coherent misinformation.

Domain-Specific Vulnerability. Vulnerability varies by domain: BM (47.2% avg IIIR) and FI (45.2%) are the highest, while CS (37.6%) is the lowest. This consistent ranking suggests certain domains contain more superficially plausible but factually incorrect web content.

The transformer attention mechanism often prioritizes textual coherence over factual accuracy, mistaking linguistic fluency for correctness. As shown in Appendix D.2 Case 3, Claude-3.7-Sonnet anchored on an authoritative Wikipedia table to incorrectly identify Zhong Shanshan as China's wealthiest person, despite retrieving newer articles favoring Zhang Yiming. Case 4 illustrates

Qwen2.5-72B-Instruct accepting "One Bangkok O4H4" as the city's tallest building; the model's attention was triggered by technical granularity (decimal heights), causing it to ignore a critical "On Hold" status qualifier. The correct answer, Magnolias Waterfront Residences, was overlooked as it lacked similar technical precision.

4.3 LLM Lacks Planning Abilities

Large language models exhibit significant deficiencies in strategic planning when orchestrating tools for multi-faceted queries. Rather than decomposing complex information needs into structured subtasks, models frequently resort to two problematic approaches. First, they employ "query compression", attempting to encapsulate multiple distinct requirements within a single, semantically overloaded search. Second, they generate redundant, poorly-structured parallel searches that fail to build upon intermediate results. Both patterns reflect an absence of coherent decomposition strategies.

Planning Deficiency. Table 1 demonstrates systematic planning deficits through Task Decomposition Deviation (TDD), where lower scores indicate closer alignment with expert strategies. Claude-3.7-Sonnet achieves the best decomposition (TDD=2.4), followed by DeepSeek-R1 (2.8) and Grok-4 (2.9), while larger open-source models show severe deviations, Llama3.1-70B-Instruct (4.7) and Qwen2.5-72B-Instruct (4.2). Technical domains exacerbate planning challenges: BM shows TDD scores of 5.3 (Llama3.1) and 4.7 (Qwen2.5) compared to 3.8 and 3.4 in CS.

As shown in subsection D.3, Case 6 exemplifies query compression: Llama3.1-70B-Instruct collapses a five-step sequential chain (list UNESCO sites $\rightarrow$ identify countries $\rightarrow$ determine religions $\rightarrow$ filter by Buddhism $\rightarrow$ retrieve unemployment rates) into a single overloaded query. Case 5 demonstrates redundant search: GPT-4o executes three overlapping simultaneous searches for ocean acidification data rather than systematically (1) establishing baseline trends, (2) gathering institution-specific reports, and (3) comparing methodologies. These cases illustrate how models fail to recognize when tasks require sequential decomposition versus parallel information gathering.

5 Mitigation Strategies

5.1 Direct Preference Optimization

To address the identified conservative search behaviors, we propose a targeted approach leveraging DPO (Algorithm 1) with systematically constructed preference data. Our training data is constructed from a corpus of queries previously discarded during BrowseBench creation phase. For each query, we generate a preference pair by producing two distinct completions with Qwen-2.5-72B-Instruct.

Preference Pair Construction. The construction of these pairs is governed by a clear heuristic: we generate one response with tool access and another without. Formally, for a query q, we obtain two responses: r_{tool} (with tool access) and r_{direct} (without tools). The preference labeling follows:

$$(r_w, r_l) = \begin{cases} (r_{\text{direct}}, r_{\text{tool}}) & \text{if } r_{\text{direct}} = r_{\text{tool}} \\ (r_{\text{tool}}, r_{\text{direct}}) & \text{if } r_{\text{tool}} \neq r_{\text{direct}} \end{cases} \quad (4)$$

where r_w denotes the chosen (preferred) response and r_l denotes the rejected response. This heuristic penalizes superfluous tool use when tools provide no new information, while rewarding tool invocation when it yields novel or more accurate content.

Following training on 10K preference pairs, Qwen-2.5-72B-Instruct exhibits more efficient tool usage behaviors. As shown in Table 2, model achieves a better balance between relying on internal weights and seeking external data, resulting in a relative reduction in USR.

5.2 Attention Refinement

To mitigate the uncritical acceptance of retrieved web content, we propose a query-aware Attention Refinement mechanism (Algorithm 2) that dynamically adjusts weights based on search intent. Our approach employs a dual-mode strategy: content directly corresponding to active search keywords receives high attention by default, while "peripheral" content receives high attention only upon an exact match. Partial or approximate matches for these peripheral constraints are masked entirely. By distinguishing between intentional and incidental retrieval, this mechanism filters out the tangentially related information that often triggers hallucinations. As shown in Table 2, this refinement enables the model to maintain focus on genuinely relevant information and improves overall retrieval accuracy.

5.3 Hierarchical Query Decomposition

To address the systematic deficiency in strategic planning, we propose a Hierarchical Query Decomposition (HQD) framework (Algorithm 3) that

Table 2: Mitigation Strategies' Effect on Qwen-2.5-72B-Instruct. M1:Direct Preference Optimization; M2:Attention Refinement; M3:Hierarchical Query Decomposition; M1+M2+M3:Hybrid mitigation strategies.

	CS			ST			BM		
Model	USR	IIIR	TDD	USR	IIIR	TDD	USR	IIIR	TDD
Qwen2.5-72B-Instruct	38.2	52.3	3.4	50.5	58.7	4.5	52.8	63.4	4.7
Δ M1	-5.2	-	-	-7.5	-	-	-8.1	-	-
Δ M2	-	-10.3	-	-	-12.8	-	-	-13.5	-
Δ M3	-	-	-1.2	-	-	-1.5	-	-	-1.6
Δ M1+M2+M3	-4.8	-12.8	-1.1	-7.0	-14.5	-1.4	-7.6	-16.1	-1.5
	EN			FI			Avg.		
Model	USR	IIIR	TDD	USR	IIIR	TDD	USR	IIIR	TDD
Qwen2.5-72B-Instruct	44.7	56.8	4.0	50.3	62.1	4.4	47.3	58.7	4.2
Δ M1	-6.8	-	-	-8.3	-	-	-7.2	-	-
Δ M2	-	-11.5	-	-	-13.7	-	-	-12.4	-
Δ M3	-	-	-1.3	-	-	-1.4	-	-	-1.4
Δ M1+M2+M3	-6.3	-12.2	-1.2	-7.8	-15.0	-1.3	-6.7	-14.1	-1.3

explicitly models query planning as a structured decision problem, enabling more effective multi-round tool coordination. Given a complex query Q, our approach leverages the LLM's inherent reasoning capabilities through carefully designed prompts to decompose it into atomic information units and construct a directed acyclic graph $G = (V, E)$, where vertices represent sub-queries $\{q_1, ..., q_n\}$ and edges encode information dependencies. Subsequently, we implement a progressive acquisition strategy that respects these dependencies while maximizing information gain per query cost: $q^* = \arg\max_{q \in \text{candidates}} \text{InfoGain}(q, I)/\text{Cost}(q)$, where information gain estimates the expected reduction in uncertainty $H(Q|I) - \mathbb{E}[H(Q|I \cup \text{Result}(q))]$. A dynamic refinement mechanism adjusts subsequent queries based on accumulated information, allowing the system to eliminate redundant searches and identify emergent information needs through in-context learning. As shown in Table 2, this approach enables the model to improve multi-round tool coordination and query decomposition.

5.4 Hybrid mitigation strategies

As shown in Table 2, when all three mitigation strategies are applied simultaneously (Algorithm 4), the model demonstrates interesting synergistic effects that diverge from simple additive expectations across different metrics. The hybrid strategy shows amplified performance on IIIR, where the improvement exceeds M2's individual contribution. This synergy arises because M3's hierarchical query decomposition provides more focused sub-queries, enabling M2's attention mechanism to more effectively filter relevant information. The structured decomposition reduces the search space for attention refinement, leading to more precise identification of truly relevant content. Conversely, the improvements in USR and TDD fall slightly short of their respective individual strategy contributions (M1 and M3). This suggests a subtle trade-off when multiple strategies operate concurrently. The enhanced information filtering from M2 and granular query decomposition from M3 inadvertently trigger more conservative search behavior, partially offsetting M1's optimization of search decision boundaries. While M1 trains the model to search more appropriately, the heightened precision requirements imposed by M2 and M3 make the model slightly more cautious in initiating searches and marginally less efficient in decision path length, as it prioritizes accuracy over brevity.

6 Conclusion

BrowseBench investigates LLM Browsing agents' over-reliance on web tools, identifying failures in redundant searching, trust calibration, and query planning. We show that interventions like DPO, AR, and HQD substantially mitigate these issues. The study emphasizes the necessity of metacognitive skills for balancing internal knowledge with retrieval, providing a framework to improve agent efficiency and information quality.

Limitations

While our work provides the first systematic investigation into browsing agent over-reliance patterns, several limitations should be noted. First, BrowseBench's controlled experimental design, though enabling rigorous analysis, may not fully capture the complexity of real-world information-seeking scenarios and dynamic web environments. Second, our proposed mitigation strategies are evaluated on specific model architectures, and their generalizability across different LLM families, scales, and rapidly evolving capabilities requires further investigation. Third, our focus on over-reliance patterns primarily addresses efficiency and error reduction but does not comprehensively examine other critical aspects such as privacy considerations, handling conflicting information sources, or robustness against adversarial content and misinformation. Additionally, determining precise boundaries of reliable internal knowledge remains challenging, particularly for information near the knowledge cutoff or in rapidly evolving domains. Finally, our evaluation metrics may not fully capture subtle trade-offs between search efficiency and information comprehensiveness—seemingly "excessive" searches might sometimes provide valuable verification or unexpected insights, suggesting the need for more nuanced frameworks that account for these complexities in future work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anthropic. 2025. Claude 3.7 sonnet. Accessed: 2025-05-20. Advanced AI model featuring: - Hybrid reasoning engine with fast/slow thinking modes, - 200K token context window (experimental), - Enhanced coding capabilities (SWE-bench Verified 70.3%), - Multi-modal text+image understanding, - Computer-use automation (preview). Released as part of the Claude 3 series on 2025-02-19".

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, and 1 others. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Chen Chen, Xinlong Hao, Weiwen Liu, Xu Huang, Xingshan Zeng, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Yuefeng Huang, and 1 others. 2025. Acebench: Who wins the match point in tool usage? *arXiv preprint arXiv:2501.12851*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, 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*.

Thibault Le Sellier de Chezelles, Maxime Gasse, Alexandre Lacoste, Massimo Caccia, Alexandre Drouin, Léo Boisvert, Megh Thakkar, Tom Marty, Rim Assouel, Sahar Omidi Shayegan, and 1 others. 2025. The browsergym ecosystem for web agent research. *Transactions on Machine Learning Research*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Feiteng Fang, Yuelin Bai, Shiwen Ni, Min Yang, Xiaojun Chen, and Ruifeng Xu. 2024. Enhancing noise robustness of retrieval-augmented language models with adaptive adversarial training. *Preprint*, arXiv:2405.20978.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025a. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Jiawei Guo, Tianyu Zheng, Yizhi Li, Yuelin Bai, Bo Li, Yubo Wang, King Zhu, Graham Neubig, Wenhu Chen, and Xiang Yue. 2025b. MAmmoTH-VL: Eliciting multimodal reasoning with instruction tuning at scale. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13869–13920, Vienna, Austria. Association for Computational Linguistics.

Shirley Kokane, Ming Zhu, Tulika Manoj Awalgaonkar, Jianguo Zhang, Akshara Prabhakar, Thai Quoc Hoang, Zuxin Liu, Rithesh RN, Liangwei Yang, Weiran Yao, and 1 others. 2025. Toolscan: A benchmark for characterizing errors in tool-use llms. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008.

Xinzhe Li. 2025. A review of prominent paradigms for llm-based agents: Tool use, planning (including rag), and feedback learning. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9760–9779.

Ziming Li, Qianbo Zang, David Ma, Jiawei Guo, Tianyu Zheng, Minghao Liu, Xinyao Niu, Yue Wang, Jian Yang, Jiaheng Liu, and 1 others. 2025. Autokaggle: A multi-agent framework for autonomous data science competitions. In *ICLR 2025 Third Workshop on Deep Learning for Code*.

Mihran Miroyan, Tsung-Han Wu, Logan King, Tianle Li, Jiayi Pan, Xinyan Hu, Wei-Lin Chiang, Anastasios N Angelopoulos, Trevor Darrell, Narges Norouzi, and 1 others. 2025. Search arena: Analyzing search-augmented llms. *arXiv preprint arXiv:2506.05334*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, and 1 others. 2022. Webgpt: Browser-assisted question-answering with human feedback, 2022. *URL https://arxiv. org/abs/2112.09332*.

OpenAI. 2024a. Gpt-4o mini: Advancing cost-efficient intelligence. Accessed: 2025-05-13.

OpenAI. 2024b. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Shishir Patil, Tianjun Wang, Yongchao Zhang, and Kurt Keutzer. 2023. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.06975*.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E Gonzalez. 2025. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, and 1 others. 2024a. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *The Twelfth International Conference on Learning Representations*.

Yujia Qin, Tianle Zhang, Yikai Zheng, Weize Chen Wang, Jure Leskovec, and Tim Callahan. 2024b. Tooleval: A multi-turn, multi-tool evaluation benchmark for large language models. *arXiv preprint arXiv:2402.00834*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

Xinxing Ren, Caelum Forder, Qianbo Zang, Ahsen Tahir, Roman J Georgio, Suman Deb, Peter Carroll, Önder Gürcan, and Zekun Guo. 2025a. Anemoi: A semi-centralized multi-agent systems based on agent-to-agent communication mcp server from coral protocol. *arXiv preprint arXiv:2508.17068*.

Xinxing Ren, Qianbo Zang, and Zekun Guo. 2025b. Simugen: Multi-modal agentic framework for constructing block diagram-based simulation models. *arXiv preprint arXiv:2506.15695*.

Evan F Risko and Sam J Gilbert. 2016. Cognitive offloading. *Trends in cognitive sciences*, 20(9):676–688.

Timo Schick, Jane Dwivedi-Yu, Silvia Dessì, Roberta Raileanu, Carlos Manzanares Lomeli, Luke Zettlemoyer, Peter West, and Wen-tau Yih. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2202.03462*.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

xAI. 2025. Grok 4. Large language model. Https://x.ai/news/grok-4.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, and 1 others. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Hongshen Xu, Zihan Wang, Zichen Zhu, Lei Pan, Xingyu Chen, Lu Chen, and Kai Yu. 2025a. Alignment for efficient tool calling of large language models. *arXiv preprint arXiv:2503.06708*.

Hongshen Xu, Zichen Zhu, Lei Pan, Zihan Wang, Su Zhu, Da Ma, Ruisheng Cao, Lu Chen, and Kai Yu. 2025b. Reducing tool hallucination via reliability alignment. In *Forty-second International Conference on Machine Learning*.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R Narasimhan. 2025. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *The Thirteenth International Conference on Learning Representations*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, and 1 others. 2024. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*.

A Prompts

Prompt for Label

You are an expert domain classifier. Your task is to classify a given query into one or more of the following domains based on its primary topic and intent.
Domain Definitions:
CS (Culture & Society): Queries related to social issues, cultural topics, arts, literature, history, politics, education, law, philosophy, psychology, linguistics, anthropology, demographics, and human behavior.
ST (Science & Technology): Queries related to computer science, engineering, mathematics, physics, chemistry, astronomy, technology products, software development, AI/ML, data science, telecommunications, and technical problem-solving.
BM (Biology & Medicine): Queries related to human health, diseases, medical treatments, pharmaceuticals, anatomy, physiology, genetics, biotechnology, nutrition, mental health, healthcare systems, and clinical research.
EN (Environment): Queries related to ecology, climate change, environmental protection, natural resources, wildlife, conservation, pollution, renewable energy, sustainability, and earth sciences.
FI (Finance): Queries related to economics, banking, investments, stock markets, cryptocurrencies, personal finance, accounting, business strategy, trade, monetary policy, and financial regulations.
Classification Guidelines: - Assign the PRIMARY domain that best matches the query's main focus - A query can belong to multiple domains if it spans multiple areas (e.g., "impact of AI on healthcare" → ST + BM) - Maximum one domain per query unless the query explicitly covers more areas - Consider the user's intent and the type of answer they seek - When uncertain between domains, choose the one most central to answering the query
Response Format:
DOMAIN(S): [Domain code(s), e.g., ST or ST, BM]
CONFIDENCE: [HIGH/MEDIUM/LOW]
REASONING: [Brief explanation (1-2 sentences) of why this domain was chosen, mentioning key words or concepts from the query that indicate this classification]
Examples:
Query: "What are the symptoms of diabetes?" DOMAIN(S): BM CONFIDENCE: HIGH REASONING: This query directly asks about a medical condition and its symptoms, which is clearly within the Biology & Medicine domain.
Query: "How does machine learning improve medical diagnosis?" DOMAIN(S): ST, BM CONFIDENCE: HIGH REASONING: This query bridges technology (machine learning) and medicine (diagnosis), requiring knowledge from both Science & Technology and Biology & Medicine domains.
Query: "What is the current interest rate set by the Federal Reserve?" DOMAIN(S): FI CONFIDENCE: HIGH REASONING: This query is about monetary policy and interest rates, which are core Finance domain topics.
Query: "How does deforestation affect biodiversity?" DOMAIN(S): EN CONFIDENCE: HIGH REASONING: This query focuses on environmental impact and ecological systems, placing it firmly in the Environment domain.
Query: "What is the historical significance of the Renaissance?" DOMAIN(S): CS CONFIDENCE: HIGH REASONING: This query asks about a cultural and historical period, which belongs to Culture & Society domain.

Now classify the following query:
Query: [query]

Prompt for Response Judge

You are an answer consistency evaluator. Your task is to determine whether Model Answer B is consistent with Reference Answer A.
Evaluation Criteria
Compare the two answers based on:
1. **Core Information**: Whether key facts, data, and conclusions match
2. **Semantic Meaning**: Whether the meaning is the same even if wording differs
3. **Completeness**: Whether B covers the main information points from A
4. **Accuracy**: Whether B contradicts any content in A

Judgment Rules
- Answer "Yes" if: B's core content aligns with A, even if expressed differently. B may include reasonable additional details or use different phrasing.
- Answer "No" if: B contradicts A factually, omits critical information, or reaches different conclusions.

Output Format
Provide your judgment in the following format:

<judgment>Yes</judgment> or <judgment>No</judgment>

Content to Evaluate
Reference Answer A:

[Content of Answer A]

Model Answer B:

[Content of Answer B]

Provide your judgment.

Prompt for Model Response

You are an expert in composing functions. You are given a question and a set of possible functions. Based on the question, you will need to make appropriate function/tool calls to achieve the purpose.

Response format:
- If no functions needed: NO_FUNCTION_NEEDED: [<answer>your answer</answer>]

- If functions needed:

[func_name1(params_name1=params_value1, params_name2=params_value2...), func_name2(params)]

- Do not include other text when making function calls

Available functions: functions

Prompt for Generation

Task: Analyze the given seed query and generate diverse variations by strategically modifying its parts of speech, temporal expressions, and linguistic tone while preserving the query's informational intent. Instructions:
1. Grammatical Analysis: First, identify the key parts of speech in the seed query:

Nouns (subjects, objects, entities)
Adjectives (descriptive modifiers)
Verbs (actions, states)
Named entities (locations, organizations, technologies, etc.)
Temporal expressions (if any)
Tone markers (formal/informal indicators)

2. Domain-Specific Transformation: Based on the query's domain, apply appropriate modifications:
Example1: Culture & Society:
Replace cultural/geographical references (e.g., American → French, Japanese, Brazilian)
Substitute cultural elements while maintaining parallel concepts Vary demographic references (age groups, social groups, etc.)

3. Temporal & Tone Augmentation:
Temporal Modifications:
Add time markers: "in 2025", "recently", "latest", "current", "upcoming", "historical"
Add temporal qualifiers: "now", "today", "this year", "in the past decade"
Add urgency indicators: "right now", "immediately", "as soon as possible"
Add frequency markers: "daily", "regularly", "occasionally"

Tone & Style Variations:
Conversational tone: Add filler words and colloquialisms
"like", "you know", "I mean", "kind of", "sort of"
"actually", "basically", "literally", "honestly"

Uncertain/questioning tone:
"maybe", "perhaps", "possibly", "probably"
"I wonder if", "does anyone know", "any idea"

4. Generation Guidelines:
Generate 8-12 diverse variations per seed query
Maintain the original query's search intent and complexity
Ensure variations reflect realistic search behaviors
Mix different augmentation strategies (don't apply all at once)
Preserve grammatical correctness and natural phrasing
Balance between formal and informal variations

Output Format:
Generated Variations: <generated>new query</generated>
Seed Query: [query]
Output:

B BrowseBench Statistics

Our dataset comprises 1,500 queries equally distributed across five domains, with each query validated against expert decision paths to ensure answerability.

Domain Distribution. The dataset contains 300 queries from each of five domains: Culture & Society (CS), Science & Technology (ST), Biology & Medicine (BM), Environment (EN), and Finance (FI). This balanced distribution ensures comprehensive evaluation across diverse knowledge areas with varying temporal dynamics and factual verification requirements.

Keyword Annotation. Each query is annotated with 3-6 keywords for fine-grained categorization. The distribution follows: 450 queries (30%) with 3 keywords, 450 queries (30%) with 4 keywords, 300 queries (20%) with 5 keywords, and 300 queries (20%) with 6 keywords. This multi-dimensional annotation enables analysis of agent behavior across different topical granularities.

Expert Baseline for USR. To establish the baseline for unnecessary search detection, we collected expert search behaviors across all queries. We categorize queries by information accessibility: 30% can be answered from the model's internal knowledge without web search, while the remaining 70% require web tool usage. Among queries requiring web tools, 46% need a single search, 39% require 2-3 searches, and 15% require 4 or more searches. On average, experts perform 3.2 ± 1.8 searches per query (calculated over queries requiring web tools), with notable domain variations: CS (2.8 ± 1.5), ST (3.6 ± 2.1), BM (3.4 ± 1.9), EN (3.1 ± 1.7), and FI (3.3 ± 1.8). The overall distribution shows that 28% of queries require 1-2 searches, 47% require 3-4 searches, and 25% require 5 or more searches.

Interference Information Detection for IIIR. To evaluate models' critical assessment capabilities, we utilize the IIIR metric to quantify the extent to which a model uncritically adopts misleading content. For each query, we identify $N_{\text{interference\_retrieved}}$ by extracting all factually incorrect claims from the URLs accessed during the search process. Our statistics show that models are exposed to an average of 5.8 ± 2.4 interfering claims per query. These claims are categorized into four types: factual errors (35%), numerical inaccuracies (28%), causal relationship misattributions (22%), and temporal errors (15%). All identified interference is strictly verified against authoritative ground truth, including academic databases and official statistics. By tracking whether these specific claims ($N_{\text{interference\_incorporated}}$) appear in the model's final output, IIIR provides a realistic benchmark for assessing a model's robustness against overtrust and its ability to filter unreliable web information.

Task Complexity for TDD. Among the 1,500 queries, 450 (30%) can be answered from internal knowledge without web tool usage, resulting in an expert decision path length of 1 step. The remaining 1,050 queries (70%) require web tools and form the effective evaluation set for TDD. These queries are stratified by task complexity: simple queries (2-3 steps, 380 queries, 36%), medium complexity queries (4-5 steps, 520 queries, 50%), and complex queries (5+ steps, 150 queries, 14%). The average expert decision path length is $L_{\text{expert}} = 3.2 \pm 1.8$ steps for queries requiring web tools. Each step represents either a `web_search`, `web_fetch` operation or a single response.

C Metrics Calculation Process

To ensure the reproducibility of our experiments and the precision of our evaluation, this section details the specific calculation workflows for the three core metrics in BrowseBench: USR, IIIR, and TDD. All automated semantic judgments (e.g., comparing answers) are performed using Qwen-2.5-7B-Instruct to maintain consistency.

C.1 Unnecessary Search Rate (USR) Calculation

The **Unnecessary Search Rate (USR)** quantifies the agent's tendency to perform redundant searches when its parametric knowledge is sufficient to answer the query correctly. The evaluation process is defined as follows:

1. **Internal Knowledge Verification (Closed-Book Check):** For every query q in the dataset, we first query the model in a "closed-book" setting (i.e., with web search tools disabled). We compare the model's direct response (A_{direct}) against the expert-annotated reference answer (A_{ref}). A judge (Qwen-2.5-7B-Instruct) evaluates whether A_{direct} provides a correct and complete answer consistent with A_{ref}. If the model answers correctly using only internal knowledge, the query is classified as *Solvable via Internal Knowledge*.

2. **Redundancy Identification:** We then observe the agent's behavior in the "tool-augmented" setting for the same query. If a query was classified as *Solvable via Internal Knowledge* but the agent still executes `web_search` or `web_fetch` operations, these search steps are marked as **unnecessary**.

3. **Calculation:** The USR is calculated as the ratio of unnecessary search actions to the total number of searches performed:

$$\text{USR} = \frac{N_{\text{unnecessary}}}{N_{\text{total}}} \times 100\% \tag{5}$$

C.2 Interference Information Incorporation Rate (IIIR) Calculation

The **Interference Information Incorporation Rate (IIIR)** measures the agent's failure to filter out irrelevant or partially matching information (Interference) and its subsequent inclusion in the final response. The process relies on strict textual relevance matching:

1. **Interference Identification (Strict Matching):** For every content segment retrieved by the agent's search tools, we assess its relevance to the specific query intent using a text matching algorithm. We compare the keywords and constraints of the query against the retrieved content.

   Criterion: If the retrieved content **does not completely match** the query's specific constraints (e.g., it matches keywords but misses the specific year, location, or context), it is labeled as *Interference Information* ($N_{\text{interference\_retrieved}}$). Any content that is not a "Perfect Match" is treated as interference for this metric.

2. **Incorporation Verification:** We analyze the agent's final generated answer (A_{final}). We check if the specific details from the labeled *Interference Information* appear in A_{final}. If the agent includes this irrelevant or mismatched information (e.g., answering with data from 2023 when the query asked for 2024 because the search result was imperfect), it is recorded as an incorporation event ($N_{\text{interference\_incorporated}}$).

3. **Calculation:**

$$\text{IIIR} = \frac{N_{\text{interference\_incorporated}}}{N_{\text{interference\_retrieved}}} \times 100\% \tag{6}$$

C.3 Task Decomposition Deviation (TDD) Calculation

Task Decomposition Deviation (TDD) evaluates the agent's planning capability by comparing its execution path length against human expert demonstrations.

1. **Step Count Extraction:**
   - **Agent Steps (L_{agent}):** We count the total number of tool invocations (accumulated `web_search` and `web_fetch` actions) performed by the agent to solve query q.
   - **Expert Steps (L_{expert}):** We retrieve the number of steps taken in the expert decision path provided in the BrowseBench dataset for the same query.
2. **Calculation:** We calculate the absolute difference between the agent's steps and the expert's steps to quantify the deviation in planning efficiency:

$$\text{TDD} = |L_{\text{agent}} - L_{\text{expert}}| \tag{7}$$

   A TDD close to 0 indicates expert-like decomposition. High positive values indicate over-decomposition (inefficient, fragmented searching), while values significantly deviating from the expert baseline suggest a lack of strategic planning.

D Case Study

D.1 Cases on LLM is a Conservative

Case 1; Model: Gemini-2.5-Pro; Domain: BM

Query: I'm not sure about this, but according to current research, what are the latest steps in the electron transport chain?

Incorrect Response: web_search("latest research electron transport chain steps")

Correct Response: I can explain the electron transport chain (ETC) based on established biochemistry knowledge - this is fundamental science that hasn't changed since it was elucidated decades ago. The electron transport chain occurs in the inner mitochondrial membrane and consists of these main steps: Complex I (NADH dehydrogenase): NADH donates electrons, which are passed through iron-sulfur clusters and flavin mononucleotide (FMN). Protons are pumped from the mitochondrial matrix to the intermembrane space. Complex II (Succinate dehydrogenase): Succinate from the citric acid cycle donates electrons through FAD. This complex doesn't pump protons but feeds electrons into the chain via coenzyme Q. Coenzyme Q (ubiquinone): This mobile carrier shuttles electrons from Complexes I and II to Complex III. Complex III (cytochrome bc1 complex): Electrons move through the Q cycle involving cytochromes b and c1. More protons are pumped across the membrane. Cytochrome c: Another mobile carrier that transfers electrons from Complex III to Complex IV. Complex IV (cytochrome c oxidase): The final complex transfers electrons to oxygen (the terminal electron acceptor), reducing it to water. Additional protons are pumped here. ATP synthase (Complex V): Uses the proton gradient created by the previous complexes to synthesize ATP from ADP and inorganic phosphate.

Analysis: This query involves fundamental biochemical knowledge that falls well within the model's training cutoff. The electron transport chain is a core concept in cellular respiration with well-established mechanisms that were elucidated decades ago. The steps of the ETC (Complex I through Complex IV, along with ATP synthase) represent stable scientific knowledge that has not changed since its discovery and characterization. The phrase "latest research" in the query is misleading - while new details about regulatory mechanisms or disease implications may emerge, the basic steps themselves are unchanging fundamental biology. Since this information has essentially zero rate of change and requires no real-time data, a direct response from the model's knowledge is both appropriate and more efficient than web search. The model should recognize that "latest steps" is a misconception, as the steps themselves are fixed, and provide the established information directly.

Case 2; Model: Kimi-K2; Domain: ST

Query: Please list 10 CNN network structures before 2022.

Incorrect Response: web_search("CNN network architecture before 2022")

Correct Response: Here are 10 CNN network structures proposed before 2022, along with their years of proposal and key features:
1. LeNet-5 (1998) - One of the earliest CNNs, designed for handwritten digit recognition (MNIST). - Key features: Convolutional layers + subsampling (pooling) + fully connected layers.
2. AlexNet (2012) - First deep CNN to win ImageNet, revolutionizing computer vision. - Key features: ReLU activation, dropout, data augmentation, GPU acceleration.
3. VGGNet (2014) - Introduced the idea of using very small (3×3) convolution filters. - Key features: Simple, deep architecture (VGG-16, VGG-19).
4. GoogLeNet/Inception-v1 (2014) - Introduced the Inception module for efficient computation. - Key features: 1×1 convolutions for dimensionality reduction, multi-scale feature extraction.
5. ResNet (2015) - Solved the vanishing gradient problem with residual connections. - Key features: Skip connections, very deep networks (up to 152 layers).
6. Inception-v3 (2015) - Improved Inception with factorized convolutions and batch normalization. - Key features: Asymmetric convolutions (e.g., 1×7 and 7×1).
7. DenseNet (2017) - Connected every layer to every other layer in a feed-forward manner. - Key features: Dense connections, parameter efficiency.
8. SENet (2017) - Introduced Squeeze-and-Excitation blocks to model channel-wise dependencies. - Key features: Attention mechanism for feature recalibration.
9. EfficientNet (2019) - Optimized scaling of depth, width, and resolution uniformly. - Key features: Compound scaling, mobile-friendly efficiency.
10. RegNet (2020) - Proposed a new network design paradigm using linear parameter growth. - Key features: Structured, reproducible architecture search.
These networks laid the foundation for modern computer vision, with innovations in depth, efficiency, and attention mechanisms.

Analysis: This query involves stable technical knowledge that falls well within the model's training cutoff. CNN architectures are foundational concepts in deep learning with well-documented specifications. Since the information requested has a slow rate of change and requires no real-time data, direct response from the model's knowledge is appropriate and more efficient than web search.

D.2 Cases on LLM Overtrusts Web Sources

Case 3; Model: Claude-3.7-Sonnet; Domain: FI

Query: In May 2025, according to Forbes, who is the wealthiest billionaire from the world's largest seafood-producing country?

Process: 1. web_search("world's largest seafood producing country 2025")
2. web_search("richest Chinese billionaire Forbes May 2025")
3. web_fetch("https://en.wikipedia.org/wiki/List_of_Chinese_by_net_worth")

WebContent: 2022 Chinese billionaires top 100 list
The hundred entrepreneurs billionaires are listed as follows, including their China (mainland) rank (C) and world rank (W), citizenship, age, net worth, and source of wealth:
C W Name Age (y.o.) Net worth (billions of USD) Source of wealth Ref.
1 17 Zhong Shanshan 77–78 Decrease 65.7 -US$3.2B The chair of Nongfu Spring
2 25 Zhang Yiming 42 Increase 50 +US$14.4B The developer of TikTok

web_fetch("https://townflex.com/top-20-richest-people-in-china")
WebContent: xxx
web_fetch("https://www.forbes.com/sites/zinnialee/2024/11/06/amid-price-war-bottled-water-billionaire-zhong-shanshan-remains-chinas-richest-person")
WebContent: xxx
web_fetch("https://technext24.com/2025/03/27/tiktok-zhang-yiming-china-richest-person") and so on

Incorrect Response: Zhong Shanshan

Correct Response: Zhang Yiming

Analysis: The model's response demonstrates systematic overtrust in seemingly authoritative sources. Zhang Yiming becomes China's richest man in March 2025, with a net worth of $57.5 billion (or $65.5 billion according to Forbes estimates), surpassing Nongfu Spring's Zhong Shanshan and Tencent's Ma Huateng.

Case 4; Model: Qwen2.5-72B-Instruct; Domain: CS

Query: What's the tallest building in Thailand's most visited city in 2024?

Process: 1. web_search("Thailand 2024 most visited city")
2. web_search("the tallest building Bangkok")
3. web_fetch("https://www.skyscrapercenter.com/building/one-bangkok-o4h4/29323")
WebContent: HEIGHT: 436.1 m / 1,431 ft
FLOORS: 92
Official Name: One Bangkok O4H4
Other Names: One Bangkok Phase 3 Tower 1
Name of Complex: One Bangkok
Type: Building
Status: On Hold
Country: Thailand
City: Bangkok
Address: Rama 4 Road
Function: Hotel / Office

web_fetch("https://www.facebook.com/groups/bkk60days/posts/1014165320820164/")
WebContent: xxx
web_fetch("https://en.wikipedia.org/wiki/List$_o f_t allest_b uildings_i n_B angkok$")$and so on$

Incorrect Response: One Bangkok O4H4

Correct Response: Magnolias Waterfront Residences Iconsiam

Analysis: One Bangkok O4H4 is still under construction, so the tallest building in Bangkok is still Magnolias Waterfront Residences Iconsiam.

D.3 Cases on LLM Lacks Planning Abilities

Case 5; Model: GPT-4o; Domain: EN

Query: Is ocean acidification accelerating or stabilizing since 2020, what do different oceanographic institutes report, and how do their measurement methodologies differ?

Incorrect Plan:
1. web_search("ocean acidification trends since 2020 oceanographic institutreports")
2. web_search("ocean acidification measurement methodologies comparisoon research institutes")
3. web_search("ocean acidification rates accelenation data")

Correct Plan:
1. Understand the overall scientific consensus on recent trends: web_search("ocean acidification trends 2020-2025")
2. Gather reports from multiple authoritative sources: web_search("NOAA ocean acidification report"), web_search("Scripps Institution of Oceanography pH measurements") and so on
3. Understand how measurement approaches vary: web_search("ocean acidification measurement methods pH alkalinity comparison") and so on

Analysis: No coherent search strategy: the queries are broad and redundant, executed simultaneously without learning from each other, trying to find ready-made comparisons rather than systematically gathering specific institutional data first and then synthesizing it.

Case 6; Model: Llama3.1-70B-Instruct; Domain: CS

Query: Which UNESCO World Heritage Sites added in the last 5 years are located in countries where the majority religion is Buddhism, and what was the unemployment rate in those countries during the year of designation?

Incorrect Plan: web_search("UNESCO sites added last 5 years Buddhism majority religion countries site designation unemployment rate")

Correct Plan: 1. List new UNESCO sites added between 2020 and 2024
2. Determine the country where each site is located
3. Query the primary religion of each country
4. Filter countries with a predominantly Buddhist religion
5. Query the unemployment rate for each country in the year the site was added

Analysis: Single query with semantic overload.

Algorithm 1 Direct Preference Optimization

Require: Query corpus $\mathcal{D}$, base model π_{base}
Ensure: Optimized policy model π_θ

Initialize preference dataset $\mathcal{P} \leftarrow \emptyset$
for each query $q \in \mathcal{D}$ **do**
  $r_{\text{tool}} \leftarrow \pi_{\text{base}}(q, \text{with\_tools})$
  $r_{\text{direct}} \leftarrow \pi_{\text{base}}(q, \text{no\_tools})$
  if $r_{\text{tool}} = r_{\text{direct}}$ **then**
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{(q, r_{\text{direct}}, r_{\text{tool}})\}$ {Penalize superfluous tool use}
  else
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{(q, r_{\text{tool}}, r_{\text{direct}})\}$ {Reward necessary tool use}
  end if
end for
Train π_θ on $\mathcal{P}$ using DPO objective
return π_θ

Algorithm 2 Attention Refinement

Require: Query Q, search keywords K_{search}, retrieved content C, other query keywords K_{other}
Ensure: Refined attention weights $\mathbf{A}$

Initialize attention weights $\mathbf{A} \leftarrow \mathbf{0}$
for each content segment $c_i \in C$ **do**
  if c_i aligns with K_{search} **then**
    $\mathbf{A}[i] \leftarrow$ high {Intentional retrieval target}
  else if c_i relates to K_{other} **then**
    if c_i **exactly matches** constraints in K_{other} **then**
      $\mathbf{A}[i] \leftarrow$ high {Complete match}
    else
      $\mathbf{A}[i] \leftarrow 0$ {Mask partial matches}
    end if
  else
    $\mathbf{A}[i] \leftarrow$ low {Irrelevant content}
  end if
end for
return $\mathbf{A}$

E Mitigation Strategies

In response to these failure modes, we develop three mitigation strategies. First, DPO(Algorithm 1) enables models to learn nuanced boundaries for search decisions. Second, Attention Refinement(Algorithm 2) helps agents focus on relevant information within retrieved content. Third, Hierarchical Query Decomposition(Algorithm 3) improves multi-round tool coordination and query decomposition. We further combine these strategies to form a hybrid strategy(Algorithm 4).

Algorithm 3 Hierarchical Query Decomposition

Require: Complex query Q, LLM $\mathcal{M}$
Ensure: Final answer A

Decompose Q into sub-queries: $G = (V, E)$ where $V = \{q_1, ..., q_n\}$
Initialize accumulated information $I \leftarrow \emptyset$
Initialize candidate queries candidates $\leftarrow \{q \in V : \text{in-degree}(q) = 0\}$
while candidates $\neq \emptyset$ and Q not fully answered **do**
  Select optimal query:
    $q^* \leftarrow \arg\max_{q \in \text{candidates}} \frac{\text{InfoGain}(q,I)}{\text{Cost}(q)}$
    where $\text{InfoGain}(q, I) = H(Q|I) - \mathbb{E}[H(Q|I \cup \text{Result}(q))]$
  Execute search q^* and retrieve $\text{Result}(q^*)$
  Update $I \leftarrow I \cup \text{Result}(q^*)$
  Refine remaining sub-queries based on I via in-context learning
  Update candidates by removing q^* and adding newly feasible queries
end while
Synthesize final answer A from accumulated information I
return A

F Ethical Statement

This research aims to improve the efficiency and reliability of LLM browsing agents, contributing to more sustainable and trustworthy AI systems. However, we acknowledge important ethical considerations: our mitigation strategies must be carefully calibrated to avoid under-reliance that could cause agents to miss critical updated information, particularly in high-stakes domains like healthcare or legal advice where outdated knowledge could cause harm. Users should be transparently informed when agents rely on training data versus real-time web information, as these sources have different reliability and recency characteristics. Additionally, improving agents' selective trust in information sources could inadvertently encode or amplify biases present in training data or web content. More efficient browsing agents may enable broader deployment of systems that mediate information access, raising concerns about user autonomy, filter bubbles, and information gatekeeping. We emphasize that our technical contributions should be accompanied by robust governance frameworks, user control mechanisms, and ongoing monitoring for unintended societal consequences. All experiments were conducted using publicly available models and datasets without collecting personal user data, and we are committed to responsible disclosure to encourage consideration of both benefits and risks of increasingly capable tool-augmented AI systems.

G Annotation Tutorial

G.1 Instructions and Task Design

The full instructions provided to the annotators are as follows:

1. **Consent and Privacy:** Before starting, all participants were informed that their responses would be anonymized and used exclusively for academic research. They were required to agree to these terms to proceed.

2. **Task Goal:** Annotators were asked to (a) verify the answerability of a given query, (b) perform a web search to find the most accurate information, and (c) record their step-by-step decision path (queries or keywords used, sources selected, and final reasoning).

3. **Quality Control:** We provided 5 standard examples to each annotator as a warm-up. Only those who achieved 100% accuracy in the warm-up were permitted to continue.

G.2 Recruitment and Compensation

Participant Selection We recruited 25 annotators through a professional data service platform. To ensure high-quality labels for specialized domains like Finance and Medicine, all participants were required to hold at least a bachelor's degree.

Compensation We ensured fair compensation. Each query was paid at an average rate of $2.0, which equates to an hourly wage of approximately $12–$15 based on the average completion time (8–10 minutes per query). This rate is significantly higher than the platform's average pay ($5/hour), reflecting the expertise required for the task.

G.3 Ethics and Safety

The queries in our dataset were manually filtered to ensure they do not contain sensitive personal information, hate speech, or harmful content. No personally identifiable information (PII) was collected from the annotators.

Algorithm 4 Hybrid Mitigation Framework for LLM Browsing Agents

Require: Query Q, base model π_{base}, training corpus $\mathcal{D}$
Ensure: Final answer A with optimized tool usage and attention

/* Phase 1: Direct Preference Optimization for Tool Usage */
Initialize preference dataset $\mathcal{P} \leftarrow \emptyset$
for each query $q \in \mathcal{D}$ **do**
  Generate $r_{\text{tool}} \leftarrow \pi_{\text{base}}(q, \text{with\_tools})$
  Generate $r_{\text{direct}} \leftarrow \pi_{\text{base}}(q, \text{no\_tools})$
  if $r_{\text{tool}} = r_{\text{direct}}$ **then**
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{(q, r_{\text{direct}}, r_{\text{tool}})\}$ {Penalize superfluous tool use}
  else
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{(q, r_{\text{tool}}, r_{\text{direct}})\}$ {Reward necessary tool use}
  end if
end for
Train optimized policy π_θ on $\mathcal{P}$ using DPO objective

/* Phase 2: Hierarchical Query Decomposition */
Decompose Q into sub-query DAG: $G = (V, E)$ where $V = \{q_1, ..., q_n\}$
Initialize accumulated information $I \leftarrow \emptyset$
Initialize candidate queries candidates $\leftarrow \{q \in V : \text{in-degree}(q) = 0\}$
while candidates $\neq \emptyset$ and Q not fully answered **do**
  Select optimal sub-query: $q^* \leftarrow \arg\max_{q \in \text{candidates}} \frac{\text{InfoGain}(q,I)}{\text{Cost}(q)}$
    where $\text{InfoGain}(q, I) = H(Q|I) - \mathbb{E}[H(Q|I \cup \text{Result}(q))]$

  /* Phase 3: Query-Aware Attention Refinement */
  Extract search keywords K_{search} from q^* and other keywords K_{other} from Q
  Execute search q^* and retrieve content C
  Initialize attention weights $\mathbf{A} \leftarrow \mathbf{0}$
  for each content segment $c_i \in C$ **do**
    if c_i aligns with K_{search} **then**
      $\mathbf{A}[i] \leftarrow$ high {Intentional retrieval target}
    else if c_i relates to K_{other} **then**
      if c_i **exactly matches** all constraints in K_{other} **then**
        $\mathbf{A}[i] \leftarrow$ high {Complete match with query requirements}
      else
        $\mathbf{A}[i] \leftarrow 0$ {Mask partial matches to prevent hallucination}
      end if
    else
      $\mathbf{A}[i] \leftarrow$ low {Irrelevant content}
    end if
  end for

  Apply attention weights $\mathbf{A}$ to filter content C
  Update $I \leftarrow I \cup \text{FilteredResult}(q^*, \mathbf{A})$
  Dynamically refine remaining sub-queries in V based on I via in-context learning
  Update candidates by removing q^* and adding newly feasible queries from V
end while

Synthesize final answer A from accumulated information I using π_θ
return A