HotelQuEST: Balancing Quality and Efficiency in Agentic Search

Guy Hadad
Ben-Gurion University
guyhada@post.bgu.ac.il

Shadi Iskander
Amazon
shadisk@amazon.com

Oren Kalinsky
Amazon
orenk@amazon.com

Sofia Tolmach
Amazon
sofiato@amazon.com

Ran Levy
Amazon
ranlevy@amazon.com

Haggai Roitman
Amazon
hroitman@amazon.com

Abstract

Agentic search has emerged as a promising paradigm for adaptive retrieval systems powered by large language models (LLMs). However, existing benchmarks primarily focus on quality, overlooking efficiency factors that are critical for real-world deployment. Moreover, real-world user queries often contain underspecified preferences, a challenge that remains largely underexplored in current agentic search evaluation. As a result, many agentic search systems remain impractical despite their impressive performance. In this work, we introduce **HotelQuEST**[1], a benchmark comprising 214 hotel search queries that range from simple factual requests to complex queries, enabling evaluation across the full spectrum of query difficulty. We further address the challenge of evaluating underspecified user preferences by collecting clarifications that make annotators' implicit preferences explicit for evaluation. We find that LLM-based agents achieve higher accuracy than traditional retrievers, but at substantially higher costs due to redundant tool calls and suboptimal routing that fails to match query complexity to model capability. Our analysis exposes inefficiencies in current agentic search systems and demonstrates substantial potential for cost-aware optimization.

1 Introduction

LLMs have enabled a new generation of autonomous agents that can navigate websites, operate tools, and assist in complex tasks (Wang et al., 2024; Zheng et al., 2024; Xie et al., 2024; Chen et al., 2024b). A key emerging application is agentic search, systems that iteratively reason, retrieve information, and synthesize answers to natural-language queries (Zhang et al., 2025a; Li et al., 2025; Han et al., 2025). In practice, search workloads vary widely: systems must process large volumes of simple queries efficiently while still handling complex, multi-hop questions that demand deeper reasoning (Suri et al., 2024).

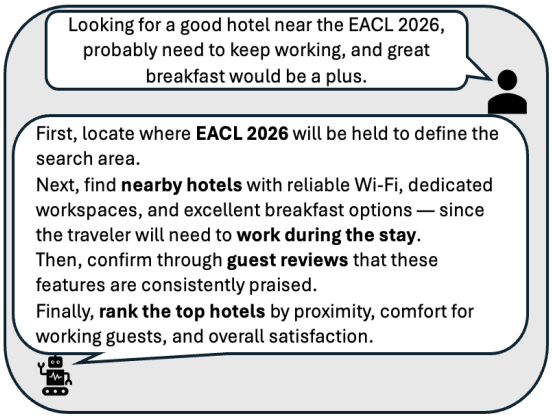

Figure 1: Illustration of a task from our benchmark.

Existing benchmarks for agentic search focus primarily on answer quality (Gou et al., 2025; Du et al., 2025), neglecting two critical dimensions for practical deployment: (i) efficiency constraints (latency, cost) that determine practical deployability (Kapoor et al., 2024), and (ii) underspecified user preferences that challenge standard relevance notions (Xi et al., 2025; Mialon et al., 2023). For instance, "dog-friendly" could mean pets are allowed for a fee, allowed freely, or only in certain areas (Choi et al., 2025). These gaps make it hard to

[1]Data and code can be found at https://github.com/amazon-science/hotel-quest-benchmark.

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 5: Industry Track), pages 209–225

judge whether agents use resources appropriately or over-compute for limited benefit.

These challenges are especially pronounced in commercial search domains like hotel booking, where queries range from simple lookups to complex, multi-hop requests with vague constraints. Consider two queries that illustrate this range: (1) "*Hotel with a gym in Berlin.*" A competent system can resolve location via filtering and match the amenity from structured attributes, without requiring multi-step reasoning. (2) "*A quiet, stroller-friendly boutique near Barcelona's center with spacious rooms and step-free access, preferably one that feels authentic and not too touristy.*" The system must combine information from both structured and unstructured sources: unstructured descriptions (e.g., "quiet," "boutique"), structured fields (room size, accessibility tags), and vague constraints like "stroller-friendly," which could imply ramps, wide corridors, or elevators.

In this paper, we introduce **HotelQuEST** (**Hotel Qu**ality & **E**fficiency **S**earch **T**estbed), a benchmark of 214 handcrafted hotel search queries, ranging from simple to complex, many of which express inherently underspecified preferences. To enable consistent and more accurate evaluation of underspecified queries, we collect clarifications – explicit statements from query authors revealing their true intent, accessible only to the judges. We jointly evaluate quality (relevance and factuality) and efficiency (cost and latency), analyze how query characteristics influence the behavior of lightweight retrievers and LLM-based agents, and establish an upper bound on achievable efficiency.

Our main contributions are:

1. A benchmark for agentic search: A set of 214 simple to complex hotel queries, each with complexity ratings, ground-truth clarifications for underspecified preferences, and structured decompositions for detailed analysis of agent behavior.

2. Joint evaluation of quality and efficiency: A systematic measurement of answer quality together with cost, token usage, and latency, capturing trade-offs between quality and practical deployability.

3. Empirical analysis exposing inefficiencies: We demonstrate that current LLM-based agents display poor cost–quality trade-offs, frequently over-investing computation for marginal quality gains. Our analysis suggests significant potential for more cost-aware agent design.

2 Related Work

2.1 Benchmarks for Agentic Search

Recent benchmarks for agentic search push beyond classical QA (Kwiatkowski et al., 2019; Ho et al., 2020) to multi-hop RAG (Tang and Yang, 2024; Yang et al., 2024; Krishna et al., 2025), and further toward multi-hop reasoning and agentic research.

In the upper section of Table 1, we summarize agent benchmarks spanning general (Gou et al., 2025; Wei et al., 2025; Mialon et al., 2023; Andrews et al., 2025), e-commerce (Yao et al., 2022), and enterprise domains (Xu et al., 2024). These works typically evaluate agents across a diverse range of tasks, involving search among other requirements, to assess their overall capabilities. The middle section of the table summarizes recent work on agentic search, highlighting that most efforts emphasize deep research (Du et al., 2025; Abaskohi et al., 2025; Rosset et al., 2025), as well as factual seeking (Xi et al., 2025) and broad search (Wong et al., 2025). However, no existing work jointly evaluates efficiency and quality, nor addresses underspecified queries where implicit user intent must be inferred – a common characteristic of real-world search that is critical for practical deployment (Kapoor et al., 2024).

2.2 Efficiency in LLMs and Agents

Recent work explores "fast" and "slow" thinking in LLMs (Kahneman, 2011; Wang et al., 2025). Slow thinking uses test-time compute to enhance reasoning (Jaech et al., 2024; Snell et al., 2024), exemplified by Chain-of-Thought (Wei et al., 2022). Although these methods deliver strong gains (Ferrag et al., 2025), they often incur computational costs that are impractical for real-world use (Feng et al., 2025). Moreover, current LLMs lack the ability to adaptively choose between these modes. Using fast thinking on complex queries degrades quality, while applying slow thinking to simple queries wastes computational resources.

Recent work proposes hybrid frameworks for adaptive mode selection (Jiang et al., 2025; Fang et al., 2025; Cheng et al., 2025), yet existing benchmarks remain limited, not specifically designed for agentic search or efficiency–quality trade-offs. With the rise of search agents (Zhang et al., 2025b), the problem has become more pronounced, as their extended reasoning traces often lead to computationally intensive processes for completing complex tasks (Xu and Peng, 2025; Li et al., 2025).

Table 1: Comparison between benchmarks. Top: agentic benchmarks involving search among other requirements across general, e-commerce, and enterprise domains. Middle: agentic search benchmarks focusing on deep research, factual seeking, and broad search. Columns **A**, **F**, and **E** indicate **Accuracy**, **Factuality**, and **Efficiency**, respectively.

Name	Domain	Size	Language	Complexity	A	F	E
Mind2Web 2 (Gou et al., 2025)	General	130	English	High	✓	✓	×
WebShop (Yao et al., 2022)	E-Commerce	12,087	English	Low	✓	×	×
BrowseComp (Wei et al., 2025)	General	1,266	English	High	✓	×	×
TheAgentCompany (Xu et al., 2024)	Enterprise	175	English	Undefined	✓	×	✓
GAIA (Mialon et al., 2023)	General assistant	466	English	Low to High	✓	×	×
GAIA2 (Andrews et al., 2025)	General	963	English	High	✓	✓	×
InfoDeepSeek (Xi et al., 2025)	Search	245	19 languages	High	✓	✓	×
DeepResearch Bench (Du et al., 2025)	Research	100	English ; Chinese	High	✓	✓	×
LiveDRBench (Java et al., 2025)	Research	100	English	High	✓	×	×
WideSearch (Wong et al., 2025)	Search	200	English ; Chinese	Medium	✓	✓	×
HotelQuEST (Ours)	Hotels	214	English	Low to High	✓	✓	✓

To the best of our knowledge, no existing benchmark systematically evaluates this capability in agentic search. Therefore, we propose a new benchmark designed to fill this gap and enable rigorous evaluation in commercial contexts.

3 The HotelQuEST Benchmark

3.1 Problem Definition

Let $\mathcal{H} = \{h_1, \ldots, h_N\}$ denote a hotel catalog. Given a natural-language query $q \in \mathcal{Q}$, we extract a finite set of *qualifiers* (constraints) $\Phi(q) = \{\varphi_1, \ldots, \varphi_m\}$ over attributes such as location, budget, amenities, etc. The task is to retrieve the top-k relevant hotels to q. For generative models, the output should include grounded evidence, which justifies the reasoning behind its selections.

3.2 Query Collection

Twenty-two human annotators participated in the data creation process, guided by a three-stage protocol designed to ensure diversity in complexity and query characteristics. An additional human reviewer then filtered out queries that did not adhere to the task guidelines, ensuring that only well-formed and goal-oriented queries were retained.

Stage 1: Query generation. Annotators wrote queries based on authentic travel scenarios they would realistically search for. We instructed them to express their requirements as they naturally would when using a natural language search interface. This yielded queries spanning simple lookups to complex and multi-constraint requests, reflecting real-world patterns where users leverage natural language interfaces rather than traditional keyword or filter-based interfaces.

Stage 2: Clarification ground truth. Each annotator also provided a **clarification**–a note that makes their underspecified assumptions explicit. This gap is evident in our query analysis and aligns with prior observations in the literature (Choi et al., 2025; Dou et al., 2007).

This step is motivated by a central insight from Thomas et al. (2024): *the only reliable "gold" relevance signal is the intent of the searcher themselves*. The goal is to capture what a capable agent must infer to correctly interpret the request. Clarifications are only available to the *judge*, and they serve as ground truth for the user's implicit intent.

Clarifications can take many forms. For example, an underspecified request like "*Hotel for a solo traveler*" is clarified as "*Find affordable hotels or hostels in safe neighborhoods suitable for solo travelers.*" Similarly, *"Hotels in London where I can see the King"* can be clarified by specifying the location being referenced, for instance, indicating that it refers to Buckingham Palace in London.

Stage 3: Complexity assessment. Annotators rated the **complexity** of each query as *Simple*, *Moderate*, or *Complex*. The annotators' complexity assessments are guided by the following three-level rubric:

- **1 = Simple:** solvable within approximately 5 minutes of search.
- **2 = Moderate:** requires roughly 5 to 15 minutes of exploration.
- **3 = Complex:** involves multi-step reasoning, cross-referencing, or multi-source search, typically exceeding 15 minutes.

This time-based interpretation of query complexity follows prior work showing that human solution time correlates with task difficulty (Gou et al., 2025), and relies on the established assumption that users can reliably self-assess the informational needs of their queries (Suri et al., 2024).

3.3 Query Characterization

Our dataset consists of **214 queries**, out of which **73.4%** include a clarification. The complexity distribution shows 37.8% are labeled Complex, 37.4% Moderate, and 24.8% Simple, providing balanced coverage across difficulty levels.

To enable fine-grained analysis of our benchmark, we decompose each query q into a set of subqueries $\{q_i\}$, where each q_i corresponds to a distinct *qualifier* capturing a specific aspect of user intent. For example:

> *"I'm going for a solo trip to San Jose, Costa Rica. Find me a hotel with great social atmosphere."*

This query contains three pairs of qualifiers: *"Solo trip"* (explicit, *Population*), *"San Jose, Costa Rica"* (explicit, *Location*), and *"Great social atmosphere"* (implicit, *Description*). We annotate each qualifier along two dimensions: **Type** (e.g., Explicit vs. Implicit, Negation) and **Content** (e.g., Location, Population, Description). This taxonomy was iteratively derived by multiple annotators analyzing an initial subset of queries (see Table 3 for the complete taxonomy with examples).

This decomposition lets us examine how query features such as the number of qualifiers, their explicitness, and content type influence model quality and efficiency across system architectures.

3.4 Hotels Corpus

We use two complementary data sources: the first is a large collection of textual *hotel descriptions* covering approximately one million hotels[2] and the second is *HotelRec* (Antognini and Faltings, 2020), a large-scale hotel recommendation dataset derived from TripAdvisor containing around 50 million user reviews. We retain only reviews corresponding to hotels for which a textual description is available. After preprocessing, we obtain **963,028** hotel descriptions. The adapted review dataset comprises **21,112,546** reviews covering **106,239** unique hotels, **18,520** cities, and **132** countries. Each hotel has **1** to **31,219** reviews, with a median of **68.0** and a mean of **198.7**. The full description of the indexing setup is presented in Appendix B.2.

4 Experimental Setup

Models. We evaluate baselines spanning the quality-efficiency spectrum: from fast, lightweight retrieval methods to sophisticated but costly LLM-based agents, for the task of returning the top-3 hotels for each query. For retrieval baselines, we employ BM25 (Lù, 2024) and top-performing embedding models from the *MTEB* benchmark (Muennighoff et al., 2023)[3] in two size categories: `all-MiniLM-L6-v2` (22M parameters) (Wang et al., 2020) and `embeddinggemma-300m` (300M parameters) (Vera et al., 2025).

As additional baselines with a reranking stage, we incorporate an LLM reranker that estimates the probability of answering "Yes" to the question of whether a given document is relevant to the query. Specifically, we employ `Qwen3-Reranker-0.6B` and `Qwen3-Reranker-4B` (Zhang et al., 2025c). Each retriever is evaluated separately on both databases, reviews and descriptions. For more details on the retrieval baselines, see Appendix B.3.

For agentic baselines, we utilize Claude models (Sonnet 4, Sonnet 3.7, and Haiku 4.5) (Anthropic, 2025a,b,c) and Qwen3-32B (Yang et al., 2025) within the LangGraph framework[4]. Each agent orchestrates three information sources: hotel *Descriptions*, customer *Reviews*, and *Web Search* via the Tavily API[5], following the iterative workflow described below.

Agentic workflow. The agent operates through an iterative process (Figure 6) for $t = 1, \ldots, T$ with memory state m_t (a textual summary of hotels retrieved so far) consisting of: *(i) Plan:* select a source $s_t \in S = \{\text{Descriptions}, \text{Reviews}, \text{Web Search}\}$ and generate a search query r_t based on the original query q and memory m_{t-1}; *(ii) Retrieve:* execute query r_t on source s_t to fetch up to k hotel candidates $H_t \subseteq \mathcal{H}$; *(iii) Filter:* prune irrelevant results from H_t and update memory to m_t with newly found hotels. The loop terminates when k hotels are identified or T has been reached, yielding the final

[2] https://www.kaggle.com/datasets/raj713335/tbo-hotels-dataset

[3] https://huggingface.co/spaces/mteb/leaderboard

[4] https://www.langchain.com/langgraph

[5] https://www.tavily.com

Section	Model	Subset	Quality		Efficiency			
			Accuracy	Factuality	Cost ($)	#Tokens	P50 (s)	P90 (s)
Retrieval only	BM25	*Reviews*	2.64	–	0.00	–	0.23	0.23
		Descriptions	1.80	–	0.00	–	0.0046	0.0046
	Dense (22M)	*Reviews*	2.56	–	0.00	–	0.0007	0.0007
		Descriptions	2.22	–	0.00	–	0.0087	0.0087
	Dense (300M)	*Reviews*	3.00	–	0.00	–	0.0054	0.0054
		Descriptions	2.63	–	0.00	–	0.0169	0.0169
Retrieval + LLM Reranker	Dense (300M) + Reranker (600M)	*Reviews*	3.26	–	0.61	–	2.9511	3.7701
		Descriptions	2.77	–	0.76	–	3.6254	4.5331
	Dense (300M) + Reranker (4B)	*Reviews*	3.32	–	3.31	–	16.070	19.7119
		Descriptions	2.96	–	4.02	–	19.2011	24.4993
LLM-based Agents	Qwen3-32B	*Full*	3.82	2.43	4.45	13M/3M	115.74	161.93
	Claude 4.5 Haiku	*Full*	3.57	2.81	18.92	1M/0.2M	69.40	155.32
	Claude 3.7 Sonnet	*Full*	4.22	2.97	96.03	14M/3.5M	364	938.42
	Claude 4 Sonnet	*Full*	4.11	2.83	50.16	7.9M/1.8M	123.44	291.76
Budget Oracle $1		*Full*	4.23	–	1.00	–	22.58	31.55
Budget Oracle $2		*Full*	4.42	–	1.94	–	32.13	44.68
Budget Oracle $4		*Full*	4.55	–	3.99	–	37.70	57.14
Quality Oracle		*Full*	4.71	–	13.10	–	62.65	127.44

Table 2: Evaluation split into **Retrieval only**, **Retrieval + LLM-based Reranker**, and **LLM-based Agents** on **Reviews** and **Descriptions**, as well as two versions of **Oracle** models. Metrics cover **Quality** and **Efficiency**.

ranked list with grounded evidence. For more details about the agent, see Appendix B.2.

Oracle models. Finally, to quantify the potential for improvement, we introduce two oracle baselines representing upper bounds on achievable quality. The **budget oracle** maximizes overall accuracy under fixed budget constraints (e.g., $1, $2, and $4), formulated as a Multiple-Choice Knapsack problem (Sinha and Zoltners, 1979). The **quality oracle** selects, per query, the cheapest model achieving the highest accuracy.

Evaluation. We evaluate the baselines along two complementary axes: *quality* and *efficiency*. For quality, we employ an LLM-as-a-judge approach to assess: (i) *accuracy*, which measures how well the answer aligns with the user's requirements, and (ii) *factuality*, which measures how well it is grounded in retrieved data with proper citations. Both metrics use a scoring guideline with well-defined criteria for assigning scores from 1 to 5, as shown in Appendix Table 5 (details in Appendix D and D.2). We use Sonnet 4.5 (Anthropic, 2025d) as the judge model. To ensure consistent evaluations and address query underspecification, we provide the LLM judge with the *Clarification* from Section 3.2, which captures the annotator's true intent. We validate this approach by measuring agreement between LLM and human evaluators on 246 answers spanning all baseline types, achieving a weighted Cohen's kappa of 0.84. For more details about agreement evaluation, see Appendix D.1.

For efficiency, we measure the total number of tokens processed (input/output), the cost of API usage[6], and latency statistics, specifically the median (**P50**) and tail (**P90**) response times. These metrics jointly capture the trade-off between model capability and practical deployability in real-world scenarios. For more details, see Appendix D.1.

5 Results and Analysis

5.1 Quality & Efficiency Comparison

Table 2 presents the main evaluation results on HotelQuEST, comparing retrieval-based baselines with LLM-based agentic systems. Retrieval methods (BM25, dense retrievers) offer near-zero cost and latency but have limited reasoning capabilities, resulting in lower overall accuracy compared to highly capable LLM-based agents. Among all models, Sonnet 3.7 achieves the highest accuracy but is also significantly more expensive (see Section 5.2 for detailed analysis and discussion).

The results reveal a substantial quality-efficiency gap: retrieval models excel in cost efficiency, while advanced LLMs lead in accuracy. This gap constrains deployment in industrial search pipelines

[6]All costs are based on Amazon Bedrock pricing as of November 2025.

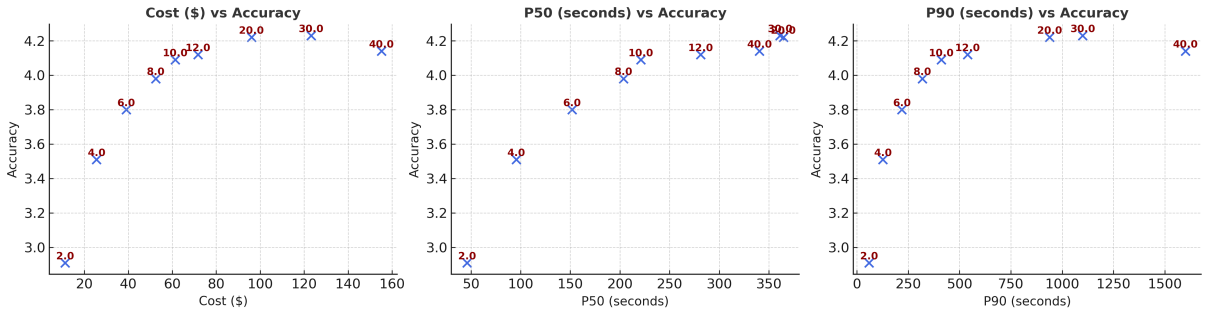

Figure 2: **Accuracy–Efficiency Trade-off.** Numbers above each point indicate the agent's iteration limit. **Left:** As cost increases, accuracy initially improves, but beyond a certain point additional cost yields no further gains. **Middle:** A similar pattern appears with median latency: accuracy rises with longer deliberation until both metrics plateau and converge. **Right:** The P90 latency curve mirrors the cost trend, indicating that on some queries the model fails to terminate early, leading to disproportionately high latency and cost.

where latency, scalability, and cost are critical, underscoring the need for efficient agentic architectures that deliver strong quality without high computational overhead.

Oracle Baselines. To establish theoretical upper bounds on routing efficiency, we evaluate two oracle strategies with perfect foresight. The budget oracle formulates model selection as a multiple-choice knapsack problem: given a global monetary budget, it selects exactly one model per query to maximize total accuracy without exceeding the budget constraint. As shown in Figure 3, accuracy exhibits a clear elbow around $2, beyond which additional expenditure yields diminishing returns.

The quality oracle operates per-query, selecting the cheapest model among those achieving the highest accuracy, thereby providing an upper bound on ideal routing given perfect knowledge of query difficulty. Figure 8 reveals that most queries achieve optimal accuracy using relatively inexpensive models, with only a small fraction requiring the most powerful agents. Both oracles demonstrate that near-optimal accuracy is attainable at a fraction of current costs: the quality oracle outperforms the best agent at lower cost, while the budget oracle at $1 achieves higher accuracy than all agents while costing 96× less than Sonnet 3.7 and 4× less than Qwen3-32B.

These results reveal substantial headroom for adaptive routing strategies and suggest that heavy agentic reasoning is rarely necessary, with large models delivering outsized benefits only on a minority of challenging queries.

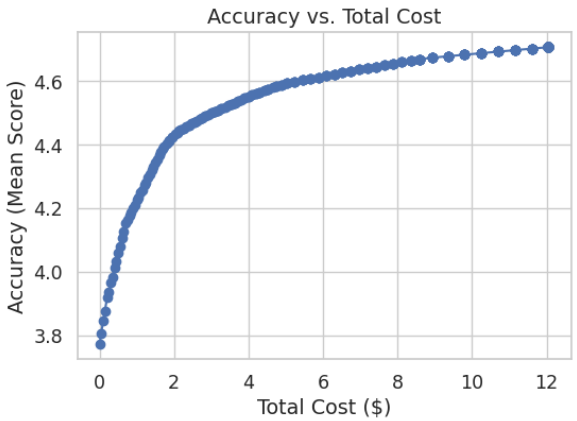

Figure 3: Budget Oracle. Accuracy achieved by solving a multiple-choice knapsack problem under varying budget limits. A clear elbow appears around $2.

5.2 Cost Inefficiency Analysis

Tracing agent reasoning trajectories reveals notable inefficiencies. Agents often continue invoking tool calls after retrieving sufficient evidence, repeatedly issuing nearly identical searches despite having access to their search history.

To quantify this, we limit the number of tool invocations for Sonnet 3.7 and measure the impact on quality and efficiency. Figure 2 shows that excessive tool calls lead to over-exploration: increased cost and latency without corresponding gains in accuracy, as median latency remains stable.

These findings highlight a critical gap: the lack of cost-aware stopping criteria in current agentic architectures. Potential solutions include contract algorithms (Shmueli-Scheuer et al., 2009), learned stopping policies (Yuan et al., 2024), and RL-based resource allocation (Aggarwal and Welleck, 2025).

5.3 Influence of Query Qualifiers

We examine how query attributes influence model quality using the taxonomy from Section 3.3, extended with query length and human-rated complexity. We apply Welch's *t*-test or Spearman correlation depending on feature type, retaining only significant results ($\alpha < 0.05$). Table 7 in Appendix C.1 presents the complete analysis across all models and query attributes.

Retrieval vs. agentic models. Query complexity significantly affects retrieval-based models but not agentic models, leading to less accurate answers that fail to fully satisfy user requirements. Query length influences both retrieval models and smaller agents like Qwen3-32B. Qwen3-32B is also sensitive to the number of qualifiers and linguistic properties like negation and subjectivity, which further decrease response accuracy. See Appendix C.1 for the full results.

Agent behavior across complexity levels. We further analyze how agents respond to query complexity. Qwen3-32B and Sonnet 4 increase cost, latency, and token usage as complexity rises, indicating that they invest more computation in harder queries. Haiku also spends more, but mainly when moving from simple to non-simple queries, with a slight cost drop at the highest level. In contrast, Sonnet 3.7 uses *less* cost, latency, and tokens as complexity increases, suggesting miscalibrated stopping behavior. Accuracy is highest on simple queries for all models and generally drops on more complex ones, with only partial recovery at the highest level. Overall, most agents respond to complexity by doing more work, but this extra effort only partially offsets the accuracy degradation on harder queries, while Sonnet 3.7 appears underinvested exactly where queries are most difficult. See Appendix C for full results.

This analysis primarily reflects the pre-retrieval stage, capturing how query properties (e.g., length, specificity, etc.) a priori affect the system's ability to retrieve relevant evidence, rather than its subsequent reasoning or answer-generation processes (Roitman, 2020).

6 Conclusion

We have introduced HotelQuEST, a benchmark designed for evaluating hotel search agents through a diverse set of manually-written queries ranging from simple to complex, often containing inherently underspecified dimensions. To mitigate ambiguity in user intent, we incorporated explicit clarifications within our evaluation framework, ensuring more reliable and interpretable evaluations. Our experiments span lightweight and cost-efficient retrieval models up to large LLM-based agents that demonstrate higher reasoning capabilities at the expense of latency and cost. We have further analyzed factors that influence model behavior in this setting, including the agent's stopping decisions and the impact of linguistic and semantic features of queries on model performance. Overall, our study highlights a critical gap between quality and efficiency, underscoring the need for future research on joint optimization strategies that balance response quality with computational and economic cost.

7 Limitations

To ensure realism and reduce annotation bias, annotators were not exposed to any specific hotels or label sets when composing queries. This design encourages natural, diverse, and unconstrained formulations. However, it also introduces uncertainty: we cannot guarantee that a single objectively optimal answer exists for every query, nor can we precisely characterize the upper bound of achievable quality.

Because large proprietary LLMs are inherently nondeterministic (Atil et al., 2024), exact reproducibility is not guaranteed. Variations in agent workflows, execution traces, and generation trajectories can lead to differences in both output quality and computational efficiency across runs.

As in other LLM-prompting studies (Chen et al., 2024a), our results may be sensitive to prompt wording and structure. Although we extensively reviewed and refined our prompts, optimizing them for this task remains an open challenge and a promising direction for future work.

Finally, similar to other human-authored query benchmarks in the field, our dataset contains a relatively limited number of queries. While this reflects the substantial cost for high-quality human annotation, it may constrain statistical power and should be considered when interpreting aggregate quality metrics.

8 Ethics Statement

During our data filtering process, we proactively removed all queries containing offensive, inappropriate, or harmful language to ensure the safety

and integrity of the dataset. Based on these procedures, we believe that the resulting benchmark poses minimal risk and is unlikely to produce negative societal impacts. All language models used in this work were accessed via the Hugging Face Hub (Wolf et al., 2020) and Amazon Bedrock. We only utilized models whose licenses explicitly permit research use, and we adhered to all relevant terms of service and usage policies throughout our experiments. We conducted our study in accordance with standard ethical principles for data handling, model usage, and reproducibility in NLP research.

References

Amirhossein Abaskohi, Tianyi Chen, Miguel Muñoz-Mármol, Curtis Fox, Amrutha Varshini Ramesh, Étienne Marcotte, Xing Han Lù, Nicolas Chapados, Spandana Gella, Christopher Pal, and 1 others. 2025. Drbench: A realistic benchmark for enterprise deep research. *arXiv preprint arXiv:2510.00172*.

Pranjal Aggarwal and Sean Welleck. 2025. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*.

Pierre Andrews, Amine Benhalloum, Gerard Moreno-Torres Bertran, Matteo Bettini, Amar Budhiraja, Ricardo Silveira Cabral, Virginie Do, Romain Froger, Emilien Garreau, Jean-Baptiste Gaya, and 1 others. 2025. Are: scaling up agent environments and evaluations. *arXiv preprint arXiv:2509.17158*.

Anthropic. 2025a. Claude 3.7 sonnet and claude code. https://www.anthropic.com/news/claude-3-7-sonnet. Accessed: 2025-10-21.

Anthropic. 2025b. Introducing claude 4. https://www.anthropic.com/news/claude-4. Accessed: 2025-5-22.

Anthropic. 2025c. Introducing claude haiku 4.5. https://www.anthropic.com/news/claude-haiku-4-5. Accessed: 2025-10-15.

Anthropic. 2025d. Introducing claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5. Accessed: 2025-9-29.

Diego Antognini and Boi Faltings. 2020. Hotelrec: a novel very large-scale hotel recommendation dataset. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4917–4923.

Berk Atil, Sarp Aykent, Alexa Chittams, Lisheng Fu, Rebecca J Passonneau, Evan Radcliffe, Guru Rajan Rajagopal, Adam Sloan, Tomasz Tudrej, Ferhan Ture, and 1 others. 2024. Non-determinism of" deterministic" llm settings. *arXiv preprint arXiv:2408.04667*.

Yue Chen, Chen Huang, Yang Deng, Wenqiang Lei, Dingnan Jin, Jia Liu, and Tat-Seng Chua. 2024a. Style: Improving domain transferability of asking clarification questions in large language model powered conversational agents. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 10633–10649.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, and 1 others. 2024b. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. *CoRR*.

Xiaoxue Cheng, Junyi Li, Zhenduo Zhang, Xinyu Tang, Wayne Xin Zhao, Xinyu Kong, and Zhiqiang Zhang. 2025. Incentivizing dual process thinking for efficient large language model reasoning. *arXiv preprint arXiv:2505.16315*.

Yoonseo Choi, Eunhye Kim, Hyunwoo Kim, Donghyun Park, Honggu Lee, Jin Young Kim, and Juho Kim. 2025. Bloomintent: Automating search evaluation with llm-generated fine-grained user intents. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*, pages 1–34.

Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the 16th international conference on World Wide Web*, pages 581–590.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. 2025. Deepresearch bench: A comprehensive benchmark for deep research agents. *arXiv preprint arXiv:2506.11763*.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. Thinkless: Llm learns when to think. *arXiv preprint arXiv:2505.13379*.

Sicheng Feng, Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. Efficient reasoning models: A survey. *arXiv preprint arXiv:2504.10903*.

Mohamed Amine Ferrag, Norbert Tihanyi, and Merouane Debbah. 2025. From llm reasoning to autonomous ai agents: A comprehensive review. *arXiv preprint arXiv:2504.19678*.

Boyu Gou, Zanming Huang, Yuting Ning, Yu Gu, Michael Lin, Weijian Qi, Andrei Kopanev, Botao Yu, Bernal Jiménez Gutiérrez, Yiheng Shu, and 1 others. 2025. Mind2web 2: Evaluating agentic search with agent-as-a-judge. *arXiv preprint arXiv:2506.21506*.

Rujun Han, Yanfei Chen, Zoey CuiZhu, Lesly Miculicich, Guan Sun, Yuanjun Bi, Weiming Wen, Hui Wan, Chunfeng Wen, Solène Maître, and 1 others. 2025. Deep researcher with test-time diffusion. *arXiv preprint arXiv:2507.16075*.

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

steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Abhinav Java, Ashmit Khandelwal, Sukruta Midigeshi, Aaron Halfaker, Amit Deshpande, Navin Goyal, Ankur Gupta, Nagarajan Natarajan, and Amit Sharma. 2025. Characterizing deep research: A benchmark and formal definition. *arXiv preprint arXiv:2508.04183*.

Lingjie Jiang, Xun Wu, Shaohan Huang, Qingxiu Dong, Zewen Chi, Li Dong, Xingxing Zhang, Tengchao Lv, Lei Cui, and Furu Wei. 2025. Think only when you need with large hybrid-reasoning models. *arXiv preprint arXiv:2505.14631*.

Daniel Kahneman. 2011. *Thinking, fast and slow*. macmillan.

Sayash Kapoor, Benedikt Stroebl, Zachary S Siegel, Nitya Nadgir, and Arvind Narayanan. 2024. Ai agents that matter. *arXiv preprint arXiv:2407.01502*.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. 2025. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4745–4759.

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.

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

Xing Han Lù. 2024. Bm25s: Orders of magnitude faster lexical search via eager sparse scoring. *arXiv preprint arXiv:2407.03618*.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836.

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*.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. Mteb: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037.

Haggai Roitman. 2020. Ictir tutorial: Modern query performance prediction: Theory and practice. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*, pages 195–196.

Corbin Rosset, Ho-Lam Chung, Guanghui Qin, Ethan Chau, Zhuo Feng, Ahmed Awadallah, Jennifer Neville, and Nikhil Rao. 2025. Researchy questions: A dataset of multi-perspective, decompositional questions for deep research. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3712–3722.

Michal Shmueli-Scheuer, Chen Li, Yosi Mass, Haggai Roitman, Ralf Schenkel, and Gerhard Weikum. 2009. Best-effort top-k query processing under budgetary constraints. In *2009 IEEE 25th International Conference on Data Engineering*, pages 928–939. IEEE.

Prabhakant Sinha and Andris A Zoltners. 1979. The multiple-choice knapsack problem. *Operations Research*, 27(3):503–515.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Siddharth Suri, Scott Counts, Leijie Wang, Chacha Chen, Mengting Wan, Tara Safavi, Jennifer Neville, Chirag Shah, Ryen W White, Reid Andersen, and 1 others. 2024. The use of generative search engines for knowledge work and complex tasks. *CoRR*.

Yixuan Tang and Yi Yang. 2024. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. In *First Conference on Language Modeling*.

Paul Thomas, Seth Spielman, Nick Craswell, and Bhaskar Mitra. 2024. Large language models can accurately predict searcher preferences. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1930–1940.

Henrique Schechter Vera, Sahil Dua, Biao Zhang, Daniel Salz, Ryan Mullins, Sindhu Raghuram Panyam, Sara Smoot, Iftekhar Naim, Joe Zou, Feiyang Chen, and 1 others. 2025. Embeddinggemma: Powerful and lightweight text representations. *arXiv preprint arXiv:2509.20354*.

Jianguo Wang, Xiaomeng Yi, Rentong Guo, Hai Jin, Peng Xu, Shengjun Li, Xiangyu Wang, Xiangzhou Guo, Chengming Li, Xiaohai Xu, and 1 others. 2021. Milvus: A purpose-built vector data management system. In *Proceedings of the 2021 international conference on management of data*, pages 2614–2627.

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.

Rui Wang, Hongru Wang, Boyang Xue, Jianhui Pang, Shudong Liu, Yi Chen, Jiahao Qiu, Derek Fai Wong, Heng Ji, and Kam-Fai Wong. 2025. Harnessing the reasoning economy: A survey of efficient reasoning for large language models. *CoRR*.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, and 1 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

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*.

Yunjia Xi, Jianghao Lin, Menghui Zhu, Yongzhao Xiao, Zhuoying Ou, Jiaqi Liu, Tong Wan, Bo Chen, Weiwen Liu, Yasheng Wang, and 1 others. 2025. Infodeepseek: Benchmarking agentic information seeking for retrieval-augmented generation. *arXiv preprint arXiv:2505.15872*.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. 2024. Travelplanner: a benchmark for real-world planning with language agents. In *Proceedings of the 41st International Conference on Machine Learning*, pages 54590–54613.

Frank F Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Z Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, and 1 others. 2024. Theagentcompany: benchmarking llm agents on consequential real world tasks. *arXiv preprint arXiv:2412.14161*.

Renjun Xu and Jingwen Peng. 2025. A comprehensive survey of deep research: Systems, methodologies, and applications. *arXiv preprint arXiv:2506.12594*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Gui, Ziran Jiang, Ziyu Jiang, and 1 others. 2024. Crag-comprehensive rag benchmark. *Advances in Neural Information Processing Systems*, 37:10470–10490.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757.

Weizhe Yuan, Ilia Kulikov, Ping Yu, Kyunghyun Cho, Sainbayar Sukhbaatar, Jason Weston, and Jing Xu. 2024. Following length constraints in instructions. *arXiv preprint arXiv:2406.17744*.

Weizhi Zhang, Yangning Li, Yuanchen Bei, Junyu Luo, Guancheng Wan, Liangwei Yang, Chenxuan Xie, Yuyao Yang, Wei-Chieh Huang, Chunyu Miao, and 1 others. 2025a. From web search towards agentic deep research: Incentivizing search with reasoning agents. *arXiv preprint arXiv:2506.18959*.

Wenlin Zhang, Xiaopeng Li, Yingyi Zhang, Pengyue Jia, Yichao Wang, Huifeng Guo, Yong Liu, and Xiangyu Zhao. 2025b. Deep research: A survey of autonomous research agents. *arXiv preprint arXiv:2508.12752*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025c. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024. Gpt-4v (ision) is a generalist web agent, if grounded. In *International Conference on Machine Learning*, pages 61349–61385. PMLR.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

A Additional Details on the HotelQuEST Benchmark

A.1 Query Length

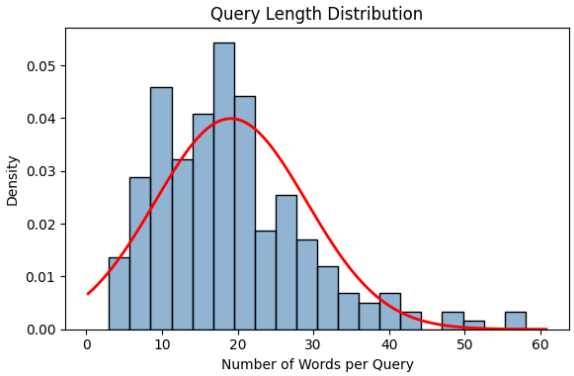

Figure 4: Distribution of query lengths.

Figure 4 presents the distribution of query lengths, shown as a histogram over the number of words per query. As observed in our analysis, query length plays a significant role, particularly for retrieval-only models, whose performance is more sensitive to shorter and less informative queries.

Qualifier Type	Qualifier Content
Explicit / Implicit	Purpose
Negation	Location
Similarity	Population
Range	Seasonality
Time-sensitive	Description
Optional / Mandatory	Rating

Table 3: Taxonomy of qualifier types and contents.

B Additional Details on the Experiments

B.1 Judgment

All judgments in this work are produced using *Claude Sonnet 4.5* (Anthropic, 2025d) as the evaluating model for his strong performance (Zheng et al., 2023). We employ two dedicated prompts: the *accuracy* prompt (see D.3) and the *factuality* prompt (see D.2). These prompts provide structured scoring criteria to ensure consistent and reproducible evaluations across all model outputs.

B.2 Agent Workflow

The agent operates with three specialized tools: one for retrieving item descriptions, one for retrieving reviews, and one for performing web search. After each tool call, the agent extracts only the information relevant to the user query and stores it in an internal notes field. This mechanism prevents repeated regeneration of long, irrelevant context across iterations and ensures that the model accumulates only the essential evidence needed for reasoning.

Figure 6 illustrates the full agentic workflow. At the beginning of each episode, the agent receives the user query and decides whether to (i) call a tool or (ii) generate a final answer. When a tool is selected, the retrieved information is summarized and added to the notes, after which the agent replans its next step. This iterative process continues until the agent determines it has sufficient evidence and produces the final answer.

Hardware. Inference latency and monetary cost are evaluated on Amazon EC2 instances. For the LLMs, we employ the Amazon Bedrock API as the serving environment. For the rerankers and retrieval components, we run all computations directly on the same EC2 machine type `g6e.4xlarge` to ensure consistent quality measurement across models.

Indexing. We construct separate vector indices for the descriptions and the reviews using *Milvus* (Wang et al., 2021) with *All-MiniLM-L6-v2* embeddings. For hotel descriptions, we adopt a *FLAT* index to enable exact similarity search, while for reviews we use an *HNSW* (Malkov and Yashunin, 2018) index to improve computational efficiency at scale.

B.3 Retrieval Baselines

For all retrieval baselines, we rely on publicly available models from the Hugging Face Hub and the `sentence-transformers` library. All embedding models and rerankers are used in their original form without additional fine-tuning. For *EmbeddingGemma*, we also adopt the prompt templates recommended by the authors to ensure consistent embedding behavior.

To index the corpora, we use *FLAT* for the hotel-description collection and *HNSW* for the reviews corpus. This choice is driven by computational constraints: the reviews corpus is too large for brute-force nearest-neighbor search, making hierarchical indexing essential for tractable retrieval. Importantly, the difference in indexing structures also explains the observed latency differences. Despite being a significantly larger corpus, the reviews collection benefits from the efficiency of *HNSW*, re-

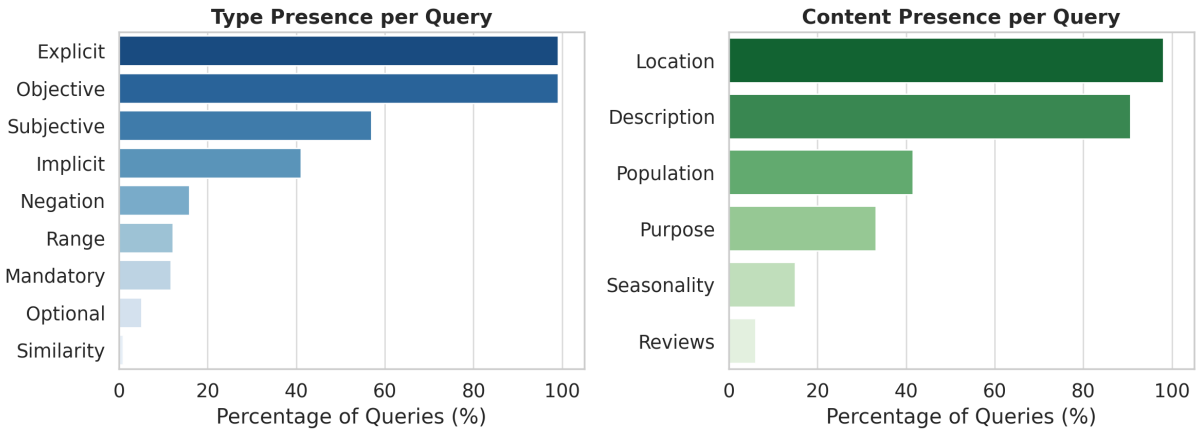

Figure 5: Analysis of the queries by the presence of qualifier attributes.

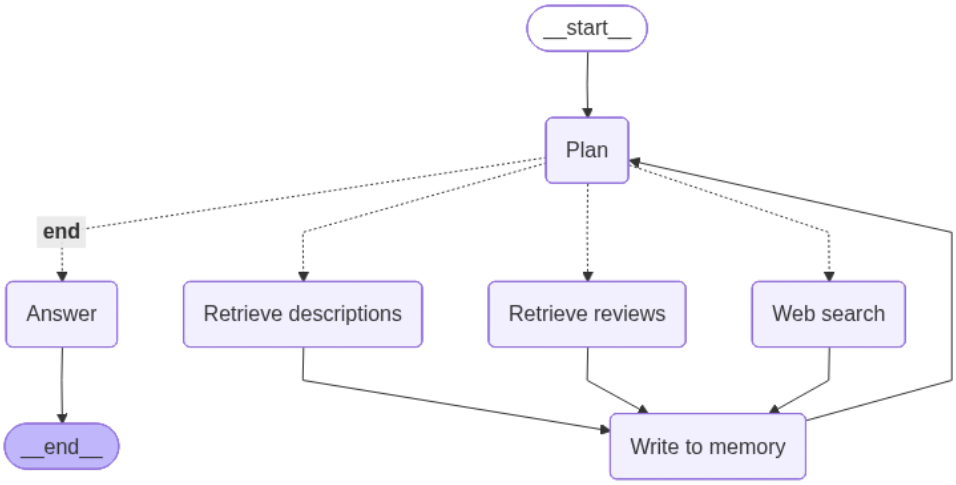

Figure 6: Illustration of the agentic workflow.

sulting in lower latency compared to FLAT. In contrast, for BM25 we observe the opposite trend—the smaller corpus yields faster retrieval, as expected under inverted-index search.

For reranking-based baselines, we first retrieve the top 100 documents from the index, apply the reranker to this candidate set, and return the top 3 documents.

All retrieval models operate under a single-batch inference setup. Consequently, the end-to-end latency for queries *without* reranking is identical across samples and is computed as:

$$\text{latency per query} = \frac{\text{batch latency}}{\text{number of queries in the batch}}.$$

This provides a consistent and fair latency comparison across all embedding-based retrieval baselines.

C Additional Analysis

C.1 Query Feature Analysis

Figure 7 reports which query features are statistically significant for each model, where a value of "1" denotes significance. The features themselves are defined in Table 3. Due to the relatively small number of queries, this analysis has certain limitations, and we exclude any feature that appears in fewer than 20% of the queries. Each feature is treated as binary, indicating whether it occurs at least once within a given query.

C.2 Quality by Complexity

We evaluate the agents within each complexity group to analyze how cost, token usage, latency, and accuracy vary as query difficulty increases. The results are presented in Table 4.

Table 4: Metrics by complexity level and model; tokens shown as inputK/outputK.

	Simple				Moderate				Complex			
Metric	Qwen3-32B	Sonnet 4	Haiku 4.5	Sonnet 3.7	Qwen3-32B	Sonnet 4	Haiku 4.5	Sonnet 3.7	Qwen3-32B	Sonnet 4	Haiku 4.5	Sonnet 3.7
Cost	0.022	0.212	0.078	0.482	0.025	0.226	0.095	0.479	0.026	0.250	0.092	0.461
Tokens	77K/18K	33K/8K	31K/8K	73K/18K	83K/20K	36K/8K	37K/10K	73K/17K	90K/21K	39K/9K	35K/10K	70K/17K
Latency (sec)	83.78	136.60	74.36	425.95	89.99	141.24	91.42	423.81	96.98	156.88	93.02	416.82
Accuracy	4.135	4.275	3.519	4.423	3.613	3.974	3.613	4.150	3.850	4.093	3.550	4.175

Score	Label	Description	Criteria
5	Exact Match	The answer completely addresses all aspects of the query with specific, actionable hotel recommendations.	• Addresses *all* requirements (location, budget, amenities, group size, etc.) • Provides *specific hotel names* and relevant details • Explains *why* each recommendation fits
4	Strong Match	Covers almost all requirements, with minor omissions or slight generalization.	• Addresses *most* requirements with relevant hotels • Missing minor detail (e.g., exact price or a less critical amenity)
3	Partial Match	Covers some requirements but misses key aspects.	• Addresses *some* requirements • May give generic advice instead of specific hotels • Missing critical constraint(s) like budget, location, or amenities
2	Weak Match	Provides tangentially relevant information but not directly aligned with query intent.	• Hotel suggestions are only loosely related • Misses multiple key requirements • Possibly recommends wrong type of property or area
1	Irrelevant	Fails to address the query requirements.	• No relevant hotel recommendations • Wrong location/context • Ignores critical constraints

Table 5: Accuracy scoring rubric for hotel recommendation answers.

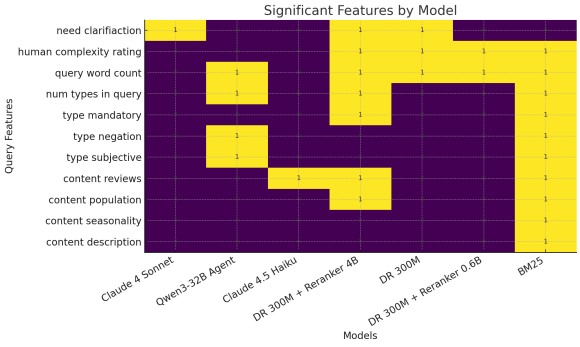

Figure 7: Analysis of query features.

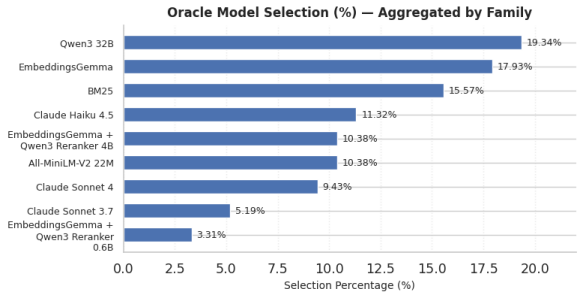

Figure 8: Quality Oracle. Distribution of selected models when choosing, for each query, the cheapest model among those achieving the highest accuracy.

D Evaluations

D.1 Human–LLM Agreement

To evaluate the reliability of our automatic scoring pipeline, we measure the alignment between hu-

man judgments and LLM-based judgments. Specifically, we analyze different aggregation setups.

Figure 9 reports the resulting confusion matrices for each aggregation scheme. The matrices demonstrate strong alignment between human annotators and the LLM evaluator, with most disagreement concentrated in borderline or partially correct cases. This suggests that the LLM-based scoring mechanism is sufficiently reliable for large-scale evaluation while remaining sensitive to nuanced differences in answer quality. In total, we manually annotated **246 answers**, covering the complete spectrum of observed model behaviors.

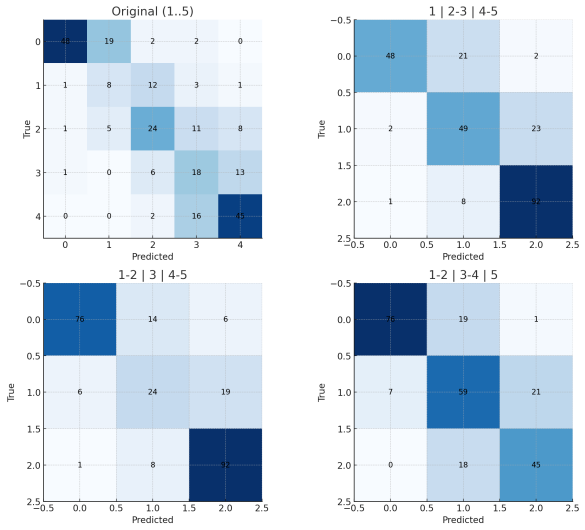

Figure 9: Confusion matrices measuring alignment between human judgments and LLM-based scoring across different levels of label aggregation.

Table 6 reports the agreement between human annotators and the LLM-as-a-judge across multiple aggregation schemes of the 1–5 rating scale. The evaluation is based on **246 answers**, each independently rated by humans for different system baselines producing varying quality scores. We assess alignment using several complementary measures: (i) *Exact Match*, capturing strict agreement; (ii) *Cohen's* κ with linear and quadratic weights, which account for partial disagreements and rating distance; and (iii) rank- and correlation-based measures, Spearman's ρ, Kendall's τ, and Pearson's r, to quantify ordinal and linear consistency. Additionally, non-parametric (*Wilcoxon*) and parametric (*paired t-test*) significance tests evaluate whether differences between distributions are statistically meaningful.

Across all aggregation schemes, correlations remain high ($\rho, r > 0.8$), and all tests indicate strong statistical significance ($p < 0.01$). This consistent alignment across diverse baselines and scoring distributions–demonstrating that the LLM-as-a-judge reliably mirrors human evaluation patterns, validating its use as a robust and scalable proxy for human judgment.

D.2 Factuality Evaluation Prompt

For the Factuality Evaluation, we use a structured prompt that includes: (i) a fixed evaluation header, (ii) a placeholder describing the type of answer being evaluated, (iii) the *User Query* and the *Model Answer*, (iv) the *Clarification* (when applicable), and (v) the complete *Context* corresponding to all hotel documents cited by the model. This context consists of the full hotel descriptions and review texts associated with every citation the agent produces, as well as any snippets retrieved through web search when the agent invokes a web tool. This setup ensures that the judge model evaluates factuality strictly based on verifiable evidence contained in the citations supplied by the agent. The full evaluation header used in the prompt is provided below.

D.3 Accuracy Evaluation Prompt

For the Accuracy Evaluation, we use a structured prompt that includes: (i) a fixed evaluation header, (ii) a placeholder describing the type of answer being evaluated, (iii) the *User Query*, (iv) the *Clarification* (when applicable), and (v) the *Model Answer*.

This prompt focuses exclusively on how well the answer satisfies the user's stated requirements, independent of factual grounding or citation quality.

The full evaluation header used in the prompt is provided below.

Setting	**Exact Match**	$\kappa_{\textbf{linear}}$	$\kappa_{\textbf{quad}}$	ρ	τ	r	**Wilcoxon** p	**t-test** p
Original (1–5)	0.5813	0.701	0.841	0.844**	0.755**	0.851**	**	**
Agg.: 1 | 2–3 | 4–5	0.7683	0.721	0.796	0.808**	0.767**	0.811**	**	**
Agg.: 1–2 | 3 | 4–5	0.7805	0.742	0.810	0.817**	0.769**	0.817**	**	**
Agg.: 1–2 | 3–4 | 5	0.7317	0.681	0.774	0.785**	0.734**	0.777**	*	*

Table 6: Agreement between human annotators and LLM-as-a-judge ratings. ** denotes $p < 0.01$; * denotes $p < 0.05$.

```
You are a Factuality Judge for HOTEL RECOMMENDATIONS.
Your goal is to assess the factual accuracy of the ANSWER strictly
based on the provided hotel descriptions and reviews.
IGNORE any outside knowledge or assumptions ,  only consider information
verifiable from the given sources.

Task: Rate how FACTUALLY ACCURATE the ANSWER is on a 1–5 scale:
1 = Completely inaccurate: contains mostly false or unsupported statements.
2 = Poor factualit: some facts are correct, but most claims lack evidence or
contradict the sources.
3 = Partially factual: roughly half the claims are supported, others are vague or unverified.
4 = Mostly factual: nearly all claims align with the sources, with only minor
inaccuracies or omissions.
5 = Fully factual: every factual statement is accurate and directly supported by a cited source.

When evaluating, consider:
• Does each factual statement about the hotel (e.g., location, amenities, ratings,
accessibility, services)
  have explicit evidence from the provided descriptions or reviews?
• Are there any hallucinated details or claims not grounded in the sources?
• Are sources cited clearly and correctly linked to each factual statement?
• Is the information consistent with the evidence, without contradictions or exaggerations?
• IMPORTANT: If any factual statement lacks an explicit source, deduct points proportionally.

Output format: Return ONLY a valid JSON object with two fields:
- score: an integer from 1 to 5
- explanation: a concise justification mentioning which parts are well-supported and which are not.

Example:
{
  "score": 4,
  "explanation": "Most details (location, breakfast, and accessibility) are supported by the
  descriptions, but the mention of a rooftop bar lacks evidence."
}

Do not include any text outside the JSON object.
```

```
You are a Relevance Judge for HOTEL RECOMMENDATIONS.
Evaluate ONLY using the provided hotel descriptions and reviews
(ignore any outside knowledge).

Task: Rate how well the ANSWER addresses the USER QUERY on a 1–5 scale:
1 = Not relevant at all: completely misses the user's needs.
2 = Slightly relevant: mentions minor aspects but not the core requirements.
3 = Moderately relevant: covers some key points but ignores important requirements.
```

```
4 = Very relevant: satisfies most requirements with only minor omissions.
5 = Perfectly relevant: fully addresses all requirements with appropriate detail.

When evaluating, consider:
• Does the answer directly address the specific hotel requirements
  (location, budget, amenities, travel dates, party size)?
• Are concrete hotel recommendations provided (hotel names + pertinent details),
  rather than generic or high-level advice?
• Is the reasoning clear, structured, and grounded in the provided descriptions/reviews?
• Are trade-offs or limitations explained when relevant?
• IMPORTANT: If the query requires recommending hotels and the answer does NOT
  provide any concrete hotel recommendation, score = 1.

Output format: Return ONLY a valid JSON object with two fields:
- score: an integer from 1 to 5
- explanation: a brief justification for the chosen score.

Example:
{
  "score": 4,
  "explanation": "The answer addresses most user requirements and provides hotel names,
  but it lacks detail about budget constraints."
}

Do not include any text outside the JSON object.
```