BrowseComp-Plus: A Fair and Disentangled Evaluation Benchmark for Deep Search Agents

Zijian Chen[*,1], **Xueguang Ma**[*,✉,1], **Shengyao Zhuang**[*,2,5], **Ping Nie**[3], **Kai Zou**[3], **Sahel Sharifymoghaddam**[1], **Andrew Liu**[1], **Joshua Green**[1], **Kshama Patel**[1], **Ruoxi Meng**[1], **Mingyi Su**[1], **Yanxi Li**[1], **Haoran Hong**[1], **Xinyu Shi**[1], **Xuye Liu**[1], **Hosna Oyarhoseini**[1], **Nandan Thakur**[1], **Crystina Zhang**[1], **Luyu Gao**[4], **Wenhu Chen**[1], **Jimmy Lin**[1]

[1] University of Waterloo [2] CSIRO [3] Independent
[4] Carnegie Mellon University [5] The University of Queensland

[*]Equal Contribution. ✉ Correspondence: x93ma@uwaterloo.ca

Abstract

Deep search agents that combine large language models with retrieval tools excel at complex, multi-hop queries. Yet, existing benchmarks such as BrowseComp rely on black-box web search APIs, facing key limitations. (1) ***Fairness***: for agents, dynamic and opaque web APIs hinder reproducibility and fair comparisons across agents. (2) ***Disentanglement***: for retrieval, the lack of a fixed document corpus makes it impossible to isolate retriever contributions from end-to-end search agent accuracy. We introduce `BrowseComp-Plus`, a benchmark derived from BrowseComp that employs a fixed, human-verified corpus, enabling controlled retrieval for deep search agents. `BrowseComp-Plus` clearly distinguishes agent performance: with a BM25 retriever, the open-source Search-R1 achieves 3.86% accuracy, while GPT-5 achieves 55.9%. Additionally, `BrowseComp-Plus` makes retrieval gains explicit: pairing GPT-5 with Qwen3-Embedding-8B retriever further improves accuracy to 70.1% while reducing search calls. Overall, `BrowseComp-Plus` provides a fair and disentangled testbed, advancing both deep search agent evaluation and retrieval research for agentic search. Code and data can be found at: https://texttron.github.io/BrowseComp-Plus/.

1 Introduction

Recent benchmarks for deep search agents, such as BrowseComp (Wei et al., 2025), demonstrate the impressive capabilities of combining large language models (LLMs) with web search tools for complex, reasoning-intensive queries. These benchmarks typically evaluate agents using black-box web search APIs to retrieve supporting documents in real time (Zhou et al., 2025; Chen et al., 2025). This design introduces critical limitations that impede fair and disentangled evaluation.

First, comparing deep search agents when evaluated using web search APIs is fundamentally unfair, due to opacity along two dimensions. **Document Corpus**: The web is highly dynamic; a benchmark's difficulty can vary substantially over time as pages are added, modified, or removed. This problem is exacerbated by search-time data contamination, where challenging benchmarks such as Humanity's Last Exam (Phan et al., 2025) leaked 3% of their answers on HuggingFace alone (Han et al., 2025). **Retrieval Algorithm:** Proprietary web search APIs do not disclose their retrieval algorithms, which may vary across providers or change over time. As we show in Section 5.2, even on the same document corpus, a better retriever can bring substantial gains, in some cases doubling agent accuracy. Together, these sources of opacity undermine the rigor of evaluation, rendering existing web-based benchmarks unreliable.

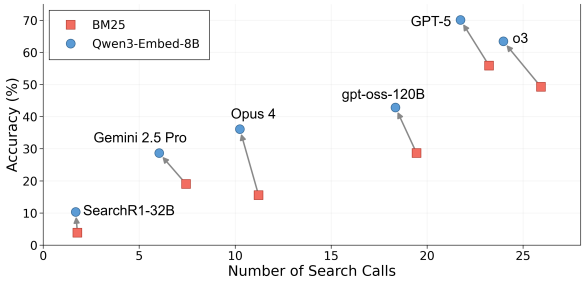

Figure 1: Accuracy vs. number of search calls for deep search agents with different retrievers. GPT-5, o3, gpt-oss are evaluated with high reasoning effort. The figure shows that **deep search agents mostly improve overall accuracy at a cost of more search calls,** whereas **better retrieval systems not only improve the overall accuracy but also reduce the number of search calls**. For reference, GPT-5 achieves 59.9% accuracy when evaluated using the Google Search API.

Second, web-based evaluation prevents disentangled analysis of retriever contributions. As discussed above, fair comparison of deep search agents requires evaluating different agents under a fixed retrieval setup. *Equally important is the converse*: evaluating different retrievers under a

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 22349–22370
July 2-7, 2026

fixed agent setup. As LLMs grow increasingly capable, retrieval researchers have begun to question whether gains from stronger retrievers are diminishing or even becoming negligible. This concern has been raised since 2022, when Gao et al. (2022) asked "Will a weak retriever theoretically suffice as the NLU and NLG models rapidly become stronger?", and has only intensified over time (Arabzadeh et al., 2025). There is a pressing need to rigorously quantify retrieval gains in modern deep search; however, such analysis is infeasible using opaque web search APIs, where it is unclear which retrieval systems are being compared.

To address these limitations, we introduce `BrowseComp-Plus`, a new benchmark that extends the original BrowseComp benchmark (Wei et al., 2025) with a fixed, human-verified document corpus, where each query is paired with explicitly identified supporting documents and hard negatives. This enables fair evaluation of deep search agents under a fixed retriever and corpus, while disentangling retriever and agent contributions. Additionally, `BrowseComp-Plus` improves reproducibility by providing a fixed retrieval environment, and enhances accessibility by replacing costly web search APIs with inexpensive local retrieval.

Using `BrowseComp-Plus`, we evaluate various open- and closed-source LLMs paired with a range of retrievers. Our results show that stronger agents improve deep search effectiveness by scaling search calls, while stronger retrievers improve both the effectiveness and efficiency of search agents. Further, we identify bottlenecks in deep search: even with state-of-the-art retrievers, agents struggle to surface all necessary evidence, and retrievers themselves exhibit substantial headroom on reasoning-intensive queries. Together, these findings motivate joint advances in agent and retrieval research, for which `BrowseComp-Plus` provides a fair and disentangled testbed.

In summary, our contributions are threefold:

- **For Agents:** By enabling controlled retrieval over a high-quality fixed corpus, we support fair and reproducible comparisons across deep search agents.
- **For Retrieval:** By disentangling and explicitly quantifying retrieval contributions in deep search, we demonstrate that retrieval remains a pivotal factor in modern deep search agents, and we reveal new findings specific to the emerging multi-turn, agentic search.
- **For the Research Ecosystem:** By releasing a shared benchmark, we place agents and retrievers on the same playing field. Retrievers can optimize directly for deep search tasks, and agents can develop against custom retrieval systems, fostering joint progress across the agent and retrieval research communities.

2 Related Work

2.1 Deep Search Agent

Deep search agents conduct tasks through iterative query reasoning, search planning, and reflection on retrieved results (Asai et al., 2024), outperforming the traditional single-round retrieval-augmented generation paradigm (Lewis et al., 2020). Commercial closed-source models such as Gemini (Gemini 2.5 Team, 2025), Opus (Anthropic Team, 2024a), and o3 (OpenAI Team, 2025b), as well as open-source models like gpt-oss (OpenAI Team, 2025a), allow access to external retrievers via tool-use APIs or MCP (Anthropic Team, 2024b). Recent research works such as Search R1 (Jin et al., 2025b) and WebSailor (Li et al., 2025), both based on the Qwen (Yang et al., 2025) model, leverage reinforcement learning to further enhance search tool capabilities. Fair evaluation of such agents, however, requires a fixed retriever system to make comparisons meaningful.

2.2 Neural Retrieval

Neural retrieval methods, such as Dense Passage Retrieval (Karpukhin et al., 2020), encode queries and documents into dense vectors using transformer models, and perform retrieval through nearest-neighbor search (Douze et al., 2024). These methods have significantly improved retrieval effectiveness compared to traditional lexical-based methods like BM25 (Robertson et al., 1994). Recent improvements in neural retrievers include advanced training strategies such as continuous pre-training (Chen et al., 2024; Gao and Callan, 2022), data augmentation (Li et al., 2023; Ma et al., 2025b; Shao et al., 2025), integration of large language models as backbones (Ma et al., 2024; Wang et al., 2024a), and LLM distillation techniques (Lee et al., 2024; Zhang et al., 2025). While retrievers are a critical component of deep search agents, the contribution of different retrievers to the overall performance of these agents remains underexplored.

2.3 Deep Search Benchmarks

Traditional benchmarks such as NaturalQuestions (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017) have significantly contributed to evaluating retrieval and retrieval-augmented generation systems (Lewis et al., 2020; Karpukhin et al., 2020; Lin et al., 2024). However, these benchmarks primarily feature single-hop questions, which typically do not require multi-step reasoning or iterative retrieval. Although datasets like HotpotQA (Yang et al., 2018) offer multi-hop questions, they are shallow in depth (2 hops), and the corpus is limited to Wikipedia. To robustly evaluate deep search agents capable of complex reasoning and iteratively retrieve many turns, benchmarks requiring deep multi-turn query interactions are essential. BrowseComp (Wei et al., 2025) stands out as a benchmark explicitly designed for this purpose, offering complex queries paired with verifiable answers. Recent extensions of BrowseComp concepts, such as BrowseComp-ZH (Zhou et al., 2025) and MedBrowseComp (Chen et al., 2025), further expand to multilingual queries and domain-specific challenges. Mind2Web2 (Gou et al., 2025) on the other hand proposes to evaluate time-varying questions with agent-as-judge.

Existing benchmarks primarily focus on question-answer evaluations of search agent systems without fixed corpora, complicating comparative assessments of retrieval methodologies. `BrowseComp-Plus` facilitates disentangled evaluations by providing a human-verified corpus, expanding the classic Cranfield paradigm (Voorhees, 2002) to modern deep search evaluation.

3 BrowseComp-Plus

3.1 Preliminaries: BrowseComp

The BrowseComp benchmark contains 1,266 challenging fact-seeking questions specifically designed to assess the capability of deep search agents to interactively and creatively navigate the web for complex, hard-to-find information (Wei et al., 2025). The questions are deliberately constructed to be difficult for both humans and LLMs, yet they feature verifiable, concise answers, enabling straightforward evaluation through simple answer matching. While widely employed for end-to-end evaluation of deep search agents with web search access, this approach forbids disentangled measurement of retrieval effectiveness within deep search.

3.2 Building the Document Corpus

Constructing a corpus for BrowseComp questions is non-trivial, presenting three key challenges:

1. **Comprehensive coverage:** The corpus must contain complete evidence to support the entire reasoning chain to answer each question.
2. **Retrieval difficulty:** The corpus should contain enough distracting negative documents so that search agents and retrievers are challenged in locating the correct evidence.
3. **Practical size:** The corpus should be large enough to yield reliable research insights, while avoiding overly large computation costs for research purposes.

To meet these criteria, we curate evidence documents through a two-stage pipeline, involving automated evidence mining followed by human verification, and mine hard-negatives via web search to attach distracting documents to each query. The sections below describe this process in detail, and present a 100k-document corpus that effectively supports the study of deep search agents.

3.2.1 Evidence Document Gathering

The original BrowseComp dataset contains only question-answer pairs. To build a document corpus with supporting evidence, the first step involves identifying relevant web pages for each question.

To achieve this, we leverage OpenAI's o3 model with web search enabled. Since the dataset intentionally makes direct retrieval of relevant documents difficult, we adopt a *reverse-engineering* strategy: We provide the answer together with the question and instruct the model to search the web for pages that have evidence supporting the answers. We also ask the model to structure the output in a table format with three columns: (1) Clue: the part of the question to address; (2) URL: the web page link containing evidence supporting the clue; and (3) Evidence: the content from the web page that supports the clue. The purpose of this table format is to facilitate human annotators in verifying each clue and its corresponding web page in the next step. The prompt is shown in Figure 4.

Of the 1,266 original question-answer pairs in BrowseComp, OpenAI o3 fails to provide supporting evidence for 124 pairs, either due to output formatting errors or because the model abstains from answering due to low confidence. For the remaining 1,142 pairs, we scrape the URLs cited

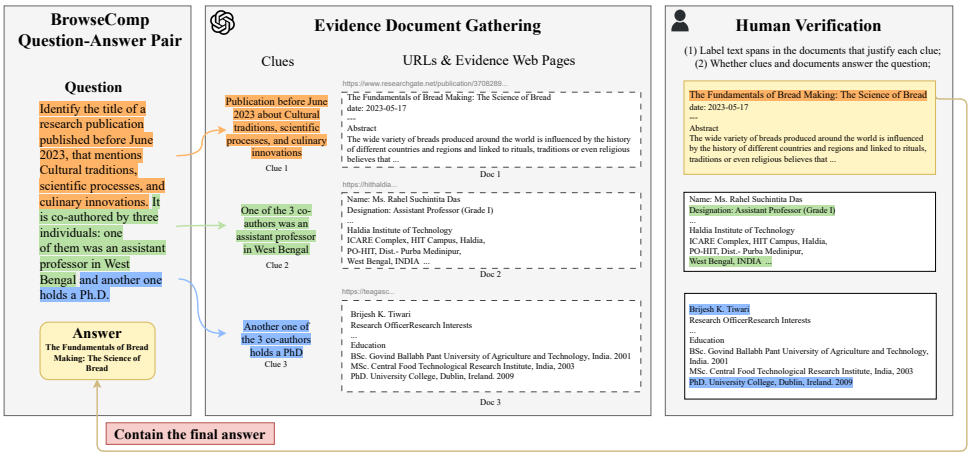

Figure 2: The two-stage pipeline of collecting evidence documents in the corpus (Section 3.2).

as evidence using Selenium,[1] and parse them with Trafilatura (Barbaresi, 2021). However, a combination of hallucinated URLs and scraping challenges prevents us from successfully scraping all of them. As a result, we exclude 137 question-answer pairs that contain at least one URL where we are unable to scrape, as missing a URL for a clue will make the question incomplete to answer.

This leaves us with 1,005 queries for the next stage: human verification.

3.2.2 Evidence Document Verification

In this stage, we aim to verify that the documents contain sufficient evidence for each clue in the questions. For each question-answer pair, we present human annotators with the output table from OpenAI o3 in the previous stage, with URLs replaced by the corresponding processed documents. Annotators are asked to:

1. Confirm that each clue is sufficiently justified by the supporting documents. Instead of simply confirming the match, annotators must label the text spans in the documents that justify each clue, as this explicit step encourages high-quality verification.

2. Determine whether the combination of clues and supporting evidence enables a human to answer the *entirety* of the question correctly. For instance, if a query asks for an individual matching five characteristics, all five must be verifiable from the documents.

If the original output from OpenAI o3 fails to meet both criteria, annotators are instructed to revise the clues and search the web for additional supporting documents for at least 20 minutes, before concluding that the desired evidence documents cannot be collected.

In addition to constructing the evidence document set, annotators also label which documents directly contain the final answer; these are designated as *gold documents*. Note that a gold document is not defined merely by containing the ground-truth answer as an exact substring; in some cases, the answer is included in the document in an implicit way. For example, a question might ask for the number of publications by a particular author, with the ground-truth answer being "7". A gold document in this case could be the author's personal webpage listing their publications; while it may not contain the string "7" explicitly, it logically contains the answer. Similarly, there are many cases where the answer appears in the document in a variant form, such as a different date format or a paraphrased phrase, rather than an exact string match. Our goal in constructing the gold document set is to provide a more robust and semantically meaningful alternative to the simple substring-based approach in identifying documents that contain the final answer.

Figure 2 illustrates the complete evidence document collection process. A detailed example, including a screenshot of the labeling interface shown to human annotators, is shown in Figure 10.

For quality control, we sample each annotator's labeled data and cross-validate them among annotators, showing over 80% agreement on average. Overall, of the 1,005 question-answer pairs from the previous stage, 830 passed human verification. The most common failure mode occurs when the documents provided by OpenAI o3 do not satisfy the two verification criteria, and human annotators are unable to gather sufficient additional evidence

[1] https://www.selenium.dev/documentation

within a reasonable effort. In addition to these, we identify and exclude several other categories of problematic cases as detailed in Appendix A.

The entire labeling process involved 14 university student annotators and required over 400 hours of manual effort. Appendix E discusses this process in more detail.

3.3 Hard Negative Mining

To ensure the collected corpus remains a reasonable size while still being challenging enough for search systems to identify correct answers among distracting documents, we mine hard negative documents via web search to form the corpus. This has proven to be effective in evaluating information retrieval systems using a sub-sampled corpus (Fröbe et al., 2025; Zhuang and Zuccon, 2022).

Specifically, we take each question from BrowseComp and prompt GPT-4o to break it down into simpler, self-contained sub-queries. On average, this results in about seven sub-queries per original query. Each sub-query is then sent to a Google Search API provider (SerpAPI), which returns up to 100 search results. We scrape these results using the same process used for collecting documents during positive example construction. We illustrate this hard negative document collection process in Figure 5. The prompt used to create these sub-queries is shown in Figure 6.

3.4 Final Corpus Statistics

After deduplicating the positive and negative documents collected, we arrive at a corpus of 100,195 documents, along with 830 queries. On average, each query contains 6.1 evidence documents, 76.28 negatives, and 2.9 gold documents. Each document averages 5179.2 words and 32296.2 characters.

4 Experimental Setup

4.1 Baselines

Search Agents We evaluate several representative commercial models with strong agentic search capabilities, ranging from the most advanced reasoning models to cost-effective ones: GPT-5, o3, GPT-4.1 (OpenAI Team, 2025b), Opus 4, Sonnet 4 (Anthropic Team, 2024a), Gemini 2.5 Pro, Gemini 2.5 Flash (Gemini 2.5 Team, 2025).

We also assess leading open-source efforts, including Qwen3-32B (Yang et al., 2025), a popular open-source reasoning LLM, and Search-R1 (Jin et al., 2025b,a), a model fine-tuned for agentic search based on the Qwen backbone. Specifically, we use the 32B checkpoint (SearchR1-32B) in (Jin et al., 2025a). Finally, we evaluate gpt-oss-120B (OpenAI Team, 2025a), a reasoning LLM optimized for search tool usage that offers multiple reasoning effort settings, ranging from low to high.

Retrievers We evaluate a range of retrievers. BM25 (Robertson et al., 1994) is the classic sparse lexical retriever. Qwen3-Embedding (Zhang et al., 2025) is a family of dense embedding retrievers, available in sizes 0.6B, 4B, and 8B, achieving state-of-the-art effectiveness on retrieval benchmarks such as MTEB (Muennighoff et al., 2023). ReasonIR (Shao et al., 2025) is a dense embedding model specifically trained for reasoning-intensive retrieval via synthetic data generation, setting a new state-of-the-art on BRIGHT (SU et al., 2025). Jina-ColBERT-v2 (Jha et al., 2024) is a late-interaction retriever that trains ColBERTv2 (Santhanam et al., 2022) from a newer BERT backbone.

We use Pyserini (Lin et al., 2021) to serve BM25, Tevatron (Ma et al., 2025a) to serve Qwen3-Embedding and ReasonIR, along with PyLate (Chaffin and Sourty, 2024) to serve Jina-ColBERT-v2.

4.2 Setup

Search Agents To perform agentic search with the LLMs, we provide the LLM with a retriever tool as tool use. We follow the original prompt from BrowseComp (Wei et al., 2025), which instructs the model to answer a given question along with a confidence estimate (expressed as a percentage). There are two revisions of the original prompts: (1) We explicitly prompt the LLM to use the provided tools to adapt to our custom search tool; (2) We instruct the model to cite the sources when generating the final answer, enabling the evaluation of citation quality. The complete prompt is shown in Figure 7. We use this prompt across all models except Search-R1, which uses the prompt aligned with its original fine-tuning.

Retrievers The retriever tool is set to retrieve the top $k = 5$ search results, where each result is truncated to the first 512 tokens of the corresponding document. This truncation is due to budget constraints, which prevent us from providing full document content. To assess the impact of this design choice, we analyze the distribution of the number of tokens required to include the ground-truth answer for each query. As illustrated in Fig-

Table 1: End-to-end performance on `BrowseComp-Plus` across LLMs and retrievers; "None" retriever denotes the parametric-only setting. All LLMs are prompted with the same tool-use prompt, except for Search-R1, which uses the prompt identical to its training. Accuracy, # Search (Search Calls issued), C.E. (Calibration Error), Recall are defined in Section 4.3. Citation Recall and Cited Ratio are defined in Section 5.4.

LLM	Retriever	Accuracy	# Search	C.E.	Recall	Citation Recall	Cited Ratio
GPT-4.1	None	3.86%	-	73.83%	-	-	-
	BM25	14.58%	10.35	68.96%	16.42%	9.16%	55.79%
	Qwen3-Embed-8B	35.42%	8.67	54.67%	36.89%	22.35%	60.59%
o3	None	19.52%	-	14.07%	-	-	-
	BM25	49.28%	25.93	12.58%	56.64%	32.37%	57.15%
	Qwen3-Embed-8B	63.49%	23.97	16.77%	73.24%	43.83%	59.84%
GPT-5	None	26.18%	-	24.57%	-	-	-
	BM25	55.90%	23.23	13.50%	61.70%	48.75%	79.01%
	Qwen3-Embed-8B	70.12%	21.74	9.11%	78.98%	61.05%	77.30%
Sonnet 4	None	1.69%	-	40.92%	-	-	-
	BM25	14.34%	9.95	29.79%	21.31%	16.25%	76.26%
	Qwen3-Embed-8B	36.75%	9.03	24.51%	47.33%	36.22%	76.53%
Opus 4	None	2.42%	-	11.95%	-	-	-
	BM25	15.54%	11.22	22.00%	22.96%	16.68%	72.65%
	Qwen3-Embed-8B	36.14%	10.24	12.79%	50.84%	36.67%	72.13%
Gemini 2.5 Flash	None	3.13%	-	79.01%	-	-	-
	BM25	15.54%	10.56	29.28%	21.45%	16.12%	75.15%
	Qwen3-Embed-8B	33.01%	9.77	21.63%	40.19%	31.29%	77.86%
Gemini 2.5 Pro	None	7.47%	-	76.72%	-	-	-
	BM25	19.04%	7.44	51.58%	22.81%	16.93%	74.22%
	Qwen3-Embed-8B	28.67%	6.04	44.08%	35.31%	24.64%	69.78%
gpt-oss-120B-high	None	3.13%	-	48.89%	-	-	-
	BM25	28.67%	19.45	46.48%	35.50%	19.69%	55.46%
	Qwen3-Embed-8B	42.89%	18.35	40.34%	52.63%	29.33%	55.73%
Qwen3-32B	None	0.96%	-	67.98%	-	-	-
	BM25	3.49%	0.92	57.41%	3.12%	2.23%	71.47%
	Qwen3-Embed-0.6B	4.10%	0.91	60.71%	3.45%	2.26%	65.51%
	Qwen3-Embed-4B	7.83%	0.89	61.06%	6.20%	4.46%	71.94%
	Qwen3-Embed-8B	10.36%	0.94	59.84%	7.80%	5.47%	70.13%
	ReasonIR	9.16%	0.91	55.15%	7.59%	5.26%	69.30%
SearchR1-32B	None	0.48%	-	-	-	-	-
	BM25	3.86%	1.78	-	2.61%	-	-
	Qwen3-Embed-0.6B	5.66%	1.73	-	5.30%	-	-
	Qwen3-Embed-4B	9.40%	1.68	-	7.90%	-	-
	Qwen3-Embed-8B	10.36%	1.69	-	10.17%	-	-
	ReasonIR	9.43%	1.74	-	8.37%	-	-

ure 3 (b), when documents are truncated to the first 512 tokens, 86.5% of queries still contain the ground-truth answer in at least one of their gold documents. Further ablations exploring alternative tool configurations are discussed in Section 5.5.

4.3 Evaluation Metrics

End-to-End Deep Search Effectiveness We report end-to-end effectiveness of the deep search agents with three metrics: Accuracy, Recall, and Search Calls. Accuracy follows BrowseComp: an LLM-as-judge (GPT-4.1) compares the model's final answer against the ground truth using the evaluation prompt shown in Figure 8. Recall measures how many human-verified evidence documents the agent retrieved during its entire interaction. Search Calls is the average number of search API invocations per query. In addition, following BrowseComp, we compute calibration error using the confidence estimates produced by the search agents, in the same way as Humanity's Last Exam (Phan et al., 2025), measuring how closely a model's predicted confidence matches the actual accuracy of its predictions. For Search-R1, we do not report calibration error because the input and output format of this model are fixed without a confidence score output. Lastly, to understand whether the accuracy obtained by each agent stems from its agentic ability or merely its parametric knowledge, we also evaluate each LLM's accuracy when directly prompted with the question, without any retriever or external knowledge.

Retrieval-Only Effectiveness For evaluating retriever effectiveness, our `BrowseComp-Plus` benchmark provides human-verified evidence documents and gold documents, along with a fixed test document collection, enabling evaluation under the Cranfield paradigm (Voorhees, 2019). Specifically, we follow standard TREC practice to create a query-document relevance label file[2] for both evidence documents and gold documents separately, and then compute Recall@k and nDCG@k to assess the effectiveness of retrievers.

5 Results

5.1 End-to-End Deep Search Agents Performance

Table 1 summarizes the overall deep search performance across different LLMs and retrievers. Proprietary models (GPT-4.1, o3, GPT-5, Sonnet 4, Opus 4, Gemini) demonstrate high answer accuracy, with OpenAI's GPT-5 achieving the highest accuracy (70.12%) when paired with the Qwen3-Embedding-8B retriever. Open-source models such as Qwen3-32B and SearchR1-32B lag behind. With the Qwen3-Embedding-8B retriever, Qwen3-32B achieves only 10.36% accuracy, compared to 35.42% for GPT-4.1 and 70.12% for GPT-5. Notably, the only high-performing open-source model we studied is gpt-oss-120B in its high reasoning mode with 42.89% accuracy, surpassing Opus 4 when both are paired with Qwen3-Embedding-8B.

In general, closed-source agents call the search tool more frequently than open-source models. For instance, OpenAI's GPT-5 and o3 issue an average of more than 20 search calls per query, while Qwen3-32B and SearchR1-32B make fewer than 2, despite being explicitly prompted to use the tool. This reflects a test-time scaling effect: more exhaustive search correlates with better outcomes and aligns with prior findings that reasoning-intensive queries benefit from exploratory retrieval. We further analyze this effect by scaling the agent's reasoning effort in Appendix B.

In the parametric-only setting where no retrieval is used ("None" retriever rows), most LLMs show very limited accuracy. Only o3 and GPT-5 perform notably better, correctly answering about 20% of the questions; this may suggest that these models were trained on BrowseComp. When comparing across different LLM agents, this potential contamination is another important factor to remember.

[2]Known as a qrel file.

Table 2: Effectiveness of retrievers. The complete question is used as the query for all retrieval methods.

Retriever	R@5	R@100	R@1000	nDCG@10
Evidence Document Retrieval				
BM25	1.2	4.7	13.7	1.6
jina-colbert-v2	5.7	18.1	35.7	7.9
Qwen3-Embed-0.6B	6.2	26.5	59.7	8.0
Qwen3-Embed-4B	9.8	40.2	71.8	14.0
Qwen3-Embed-8B	14.5	47.7	76.7	20.3
ReasonIR-8B	12.2	43.6	73.9	16.8
Gold Document Retrieval				
BM25	1.4	6.1	17.3	1.7
jina-colbert-v2	6.6	20.4	39.7	6.8
Qwen3-Embed-0.6B	8.5	30.5	66.2	7.4
Qwen3-Embed-4B	13.0	47.3	77.0	13.6
Qwen3-Embed-8B	18.5	55.8	83.5	19.5
ReasonIR-8B	15.3	49.7	78.9	15.5

5.2 Effect of Retrieval Quality

First, we evaluate retriever effectiveness in a retrieval-only setting: Table 2 reports results when the original full queries are given directly to the retriever. Relative to BM25, Qwen3-Embedding-8B and ReasonIR-8B achieve substantially higher recall and nDCG for both evidence document retrieval and gold document retrieval. Notably, within the Qwen3 embedding family, we observe a clear model size scaling law, where larger models consistently perform better.

When retrievers are paired with agents, a consistent trend emerges across all agents: stronger retrieval leads to substantially higher accuracy. As shown in Table 1, replacing BM25 with Qwen3-Embedding-8B can more than double the accuracy for Sonnet 4 and Opus 4. Even for particularly strong agents like GPT-5, we still see meaningful accuracy gains from 55.9% to 70.12%.

Stronger retrievers also reduce the number of search calls. For most proprietary models, Qwen3-Embedding-8B reduces search calls by approximately 1–3 compared to BM25. That is, better retrieval not only improves effectiveness, but also efficiency. This search turn reduction directly translates into lower Agent API costs; as shown in Table 9, agents using Qwen3-Embedding-8B consistently incur lower costs due to fewer input and output tokens.

Beyond first-stage retrieval, Appendix C analyzes the effect of rerankers, showing the potential of further gains for deep search agents.

5.3 Oracle Retrieval

We evaluate effectiveness in an extreme oracle setting, where search agents are prompted with all labeled positive documents to answer the ques-

tions. In this setup, GPT-4.1 achieves an accuracy of 93.49%. This highlights two key points. First, it showcases the importance of the retriever: if the retriever is of perfect quality, search agents can attain substantially high accuracy on complex reasoning tasks in `BrowseComp-Plus`, in contrast to the 14.58% baseline accuracy of GPT-4.1 when using BM25 as the retriever. Second, it validates the quality of the `BrowseComp-Plus` corpus itself: GPT-4.1, a non-reasoning model, is able to correctly answer 93.49% of questions using only the evidence documents in the corpus. For the remaining 6.51% of cases, human annotators reviewed each instance and confirmed that the answers are indeed answerable from the positive documents; the errors stem solely from GPT-4.1's failure to reason correctly.

5.4 Citation Analysis

Although BrowseComp queries require short answers for reliable verification, BrowseComp's setup prompts all agents to produce a long-form explanation in addition to their concise final answer. We further extend the prompt so that agents provide citations in their explanations. Combined with our evidence document labels, this enables us to analyze citation quality.

In Table 1, Citation Recall measures the recall of cited documents against the evidence documents, while Recall measures the recall of all documents retrieved by the agent via search. In comparison, Citation Recall is consistently lower, indicating that agents often fail to identify that some retrieved documents are useful and should be cited.

This is illustrated by comparing Opus 4 and gpt-oss-120B-high, both using the Qwen3-Embedding-8B retriever. While their search Recall (50.84% vs. 52.63%) is similar, gpt-oss-120B-high achieves higher accuracy (36.14% vs. 42.89%), whereas Opus 4 attains higher Citation Recall (36.67% vs. 29.33%). This effect is highlighted in the Cited Ratio column, measuring the fraction of retrieved evidence documents that are ultimately cited.

Overall, `BrowseComp-Plus` evaluates citation quality as an additional axis of effectiveness for deep search agents; high-quality citations improve verifiability and trustworthiness for human users beyond accuracy alone.

5.5 Effect of Document Reading Strategy

In previous experiments, we always present only the first 512 tokens of each retrieved document as a preview to the LLM during each round of search and reasoning, due to token budget constraints. However, in realistic deep search scenarios, agents often have access to a document reader tool that enables reading the full content of a document. To evaluate the potential benefit of such a tool, we conduct experiments with GPT-4.1 and Qwen3-32B, both with and without access to a whole-document reader (referred to as the get-document tool). Figure 9 contains the revised prompt used when the get-document tool is added.

Table 3: Comparison of Qwen3-32B and GPT-4.1 with get-document tool, using Qwen3-Embedding-8B as retriever. C.E. denotes Calibration Error.

Model	Accuracy	# Search	# Get-Doc	C.E.
GPT-4.1	35.42%	8.67	N/A	54.67%
+ get-doc	43.61%	10.03	1.85	54.28%
Qwen3-32B	10.36%	0.94	N/A	59.84%
+ get-doc	11.69%	1.01	0.27	56.47%

Results are shown in Table 3. For GPT-4.1, enabling the get-document tool improves accuracy from 35.42% to 43.61%, with a modest increase in search calls (from 8.67 to 10.03) and an average of 1.85 full-document reads per query, confirming that full-document access provides additional useful context that enhances decision-making. For Qwen3-32B, which performs worse overall, the benefit is more modest. Accuracy improves slightly from 10.36% to 11.69%, and the number of get-document calls remains low (0.27 per query on average). This suggests that while the tool can help, the model's limited tool-use ability constrains its ability to exploit the additional information.

6 Conclusion

We introduce `BrowseComp-Plus`, a benchmark to address the fairness and disentanglement challenges in evaluating deep search agents. By grounding each query in a fixed, human-verified corpus, `BrowseComp-Plus` enables controlled assessment of retrieval and agent components. For agents, we improve evaluation rigor and reproducibility. For retrievers, we make evaluation possible, and demonstrate that they substantially benefit both the effectiveness and efficiency of deep search agents.

Search agents combine search and agent efforts, yet prior works that rely on opaque web APIs obscure the role of search. By explicitly naming the retrieval component, we encourage future research in this area with retrieval in mind.

For agents, while we focus on retriever impacts during inference, a promising future direction is to study the role of retrievers during agent optimization. For instance, training an agent with a weaker retriever may be more difficult, but would this scarcity during training make the agent more capable once trained? This also relates to broader questions about "out-of-distribution" tool-use, such as how well an agent trained with BM25 generalizes to embedding-based retrievers at inference, and how such generalization can be improved. Understanding how retriever quality shapes the agent's learning dynamics remains an open question.

For retrievers, `BrowseComp-Plus` introduces a new task. As agents increasingly replace humans as the primary consumer of search, retrieval should be evaluated and optimized in the context of agents, instead of solely on traditional benchmarks like BEIR (Thakur et al., 2021). We show that retrieval continues to be important in search agents, and reveal that substantial headroom exists through our oracle experiment. To practitioners, better retrieval not only improves the same agent's accuracy and reduces token costs, but also opens the option of pairing a cheaper agent with better retrieval to match orders-of-magnitude larger models, yielding substantial real-world benefits.

Overall, `BrowseComp-Plus` serves as an ideal testbed for pursuing these directions, enabling systematic and fine-grained analyses of agent-retriever interactions for deep search. By releasing our benchmark and baselines, we aim to catalyze the next generation of deep search agents.

Limitations

`BrowseComp-Plus` has several limitations that we acknowledge and hope future work can address. First, although the corpus is constructed through careful human verification, we cannot guarantee the absence of false negatives, where documents contain relevant information but are not labeled as evidence. This limitation is present in all large-scale information retrieval corpora, since exhaustively judging every document in a large corpus is infeasible, and is generally accepted as a tradeoff for attempting to mimic web-scale retrieval (Fröbe et al., 2025); nevertheless, we acknowledge that false negatives may be present in `BrowseComp-Plus`, potentially introducing a gap between our benchmark and the most ideal evaluation setting. Second, the initial evidence-gathering step uses an OpenAI model (o3) to propose candidate URLs, which may introduce bias toward distributions that are more easily surfaced by that model; although humans subsequently edited or replaced many documents, this potential bias should be noted. Third, `BrowseComp-Plus` primarily evaluates textual evidence and does not fully capture the diversity of real-world web content, such as interactive pages, dynamic layouts, multimedia, or unparsed PDFs. Finally, in this work we focus on evaluation based on short, conclusive answers and cited documents within long-form responses. Comprehensive evaluation of generated reports for complex, ambiguous tasks remains an open direction for future work.

Ethical Considerations

The `BrowseComp-Plus` dataset extends OpenAI's BrowseComp, which is released under the MIT license. The augmented corpus was obtained by scraping documents from publicly accessible web sources searched via a Google API provider. As the data is drawn solely from open web content, we assess the ethical and legal risks to be minimal.

Acknowledgments

This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Anthropic Team. 2024a. The Claude 3 model family: Opus, Sonnet, Haiku.

Anthropic Team. 2024b. Introducing the model context protocol.

Negar Arabzadeh, Ziheng Chen, Fabio Petroni, Federico Siciliano, Fabrizio Silvestri, and Giovanni Trappolini. 2025. IR-RAG @SIGIR25: The second edition of the workshop on information retrieval's role in RAG systems. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '25, page 4168–4171, New York, NY, USA. Association for Computing Machinery.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

Adrien Barbaresi. 2021. Trafilatura: A Web Scraping Library and Command-Line Tool for Text Discovery and Extraction. In *Proceedings of the Joint*

Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations, pages 122–131. Association for Computational Linguistics.

Antoine Chaffin and Raphaël Sourty. 2024. PyLate: Flexible training and retrieval for late interaction models.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. BGE M3-Embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv:2402.03216*.

Shan Chen, Pedro Moreira, Yuxin Xiao, Sam Schmidgall, Jeremy Warner, Hugo Aerts, Thomas Hartvigsen, Jack Gallifant, and Danielle S. Bitterman. 2025. MedBrowseComp: Benchmarking medical deep research and computer use. *arXiv:2505.14963*.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The Faiss library. *arXiv:2401.08281*.

Maik Fröbe, Andrew Parry, Harrisen Scells, Shuai Wang, Shengyao Zhuang, Guido Zuccon, Martin Potthast, and Matthias Hagen. 2025. Corpus subsampling: Estimating the effectiveness of neural retrieval models on large corpora. In *Advances in Information Retrieval: 47th European Conference on Information Retrieval, ECIR 2025, Lucca, Italy, April 6–10, 2025, Proceedings, Part I*, page 453–471, Berlin, Heidelberg. Springer-Verlag.

Luyu Gao and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2843–2853, Dublin, Ireland. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2022. Precise zero-shot dense retrieval without relevance labels. *arXiv:2212.10496*.

Gemini 2.5 Team. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv:2507.06261*.

Boyu Gou, Zanming Huang, Yuting Ning, Yu Gu, Michael Lin, Botao Yu, Andrei Kopanev, Weijian Qi, Yiheng Shu, Jiaman Wu, Chan Hee Song, Bernal Jimenez Gutierrez, Yifei Li, Zeyi Liao, Hanane Nour Moussa, TIANSHU ZHANG, Jian Xie, Tianci Xue, Shijie Chen, and 7 others. 2025. Mind2Web 2: Evaluating agentic search with agent-as-a-judge. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Ziwen Han, Meher Mankikar, Julian Michael, and Zifan Wang. 2025. Search-time data contamination. *arXiv:2508.13180*.

Rohan Jha, Bo Wang, Michael Günther, Georgios Mastrapas, Saba Sturua, Isabelle Mohr, Andreas Koukounas, Mohammad Kalim Wang, Nan Wang, and Han Xiao. 2024. Jina-ColBERT-v2: A general-purpose multilingual late interaction retriever. In *Proceedings of the Fourth Workshop on Multilingual Representation Learning (MRL 2024)*, pages 159–166, Miami, Florida, USA. Association for Computational Linguistics.

Bowen Jin, Jinsung Yoon, Priyanka Kargupta, Sercan O. Arik, and Jiawei Han. 2025a. An empirical study on reinforcement learning for reasoning-search interleaved LLM agents. *arXiv:2505.15117*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025b. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv:2503.09516*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Jinhyuk Lee, Zhuyun Dai, Xiaoqi Ren, Blair Chen, Daniel Cer, Jeremy R. Cole, Kai Hui, Michael Boratko, Rajvi Kapadia, Wen Ding, Yi Luan, Sai Meher Karthik Duddu, Gustavo Hernandez Abrego, Weiqiang Shi, Nithi Gupta, Aditya Kusupati, Prateek Jain, Siddhartha Reddy Jonnalagadda, Ming-Wei Chang, and Iftekhar Naim. 2024. Gecko: Versatile text embeddings distilled from large language models. *arXiv:2403.20327*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020.

Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

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

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv:2308.03281*.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 2356–2362, New York, NY, USA. Association for Computing Machinery.

Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Richard James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, Luke Zettlemoyer, and Wen tau Yih. 2024. RA-DIT: Retrieval-augmented dual instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Wenhan Liu, Xinyu Ma, Weiwei Sun, Yutao Zhu, Yuchen Li, Dawei Yin, and Zhicheng Dou. 2025. ReasonRank: Empowering passage ranking with strong reasoning ability. *arXiv:2508.07050*.

Xueguang Ma, Luyu Gao, Shengyao Zhuang, Jiaqi Samantha Zhan, Jamie Callan, and Jimmy Lin. 2025a. Tevatron 2.0: Unified document retrieval toolkit across scale, language, and modality. SIGIR '25, page 4061–4065, New York, NY, USA. Association for Computing Machinery.

Xueguang Ma, Xi Victoria Lin, Barlas Oguz, Jimmy Lin, Wen-tau Yih, and Xilun Chen. 2025b. DRAMA: Diverse augmentation from large language models to smaller dense retrievers. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30170–30186, Vienna, Austria. Association for Computational Linguistics.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2024. Fine-tuning llama for multi-stage text retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 2421–2425, New York, NY, USA. Association for Computing Machinery.

Xueguang Ma, Xinyu Zhang, Ronak Pradeep, and Jimmy Lin. 2023. Zero-shot listwise document reranking with a large language model. *arXiv:2305.02156*.

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, Dubrovnik, Croatia. Association for Computational Linguistics.

OpenAI Team. 2025a. GPT-OSS-120B & 20B model card.

OpenAI Team. 2025b. OpenAI o3 and o4-mini system card.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The FineWeb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, Michael Choi, Anish Agrawal, Arnav Chopra, Adam Khoja, Ryan Kim, Richard Ren, Jason Hausenloy, Oliver Zhang, Mantas Mazeika, and 1090 others. 2025. Humanity's Last Exam. *arXiv:2501.14249*.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference, TREC 1994, Gaithersburg, Maryland, USA, November 2-4, 1994*, volume 500-225 of *NIST Special Publication*, pages 109–126. National Institute of Standards and Technology (NIST).

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and efficient retrieval via lightweight late interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3715–3734, Seattle, United States. Association for Computational Linguistics.

Rulin Shao, Rui Qiao, Varsha Kishore, Niklas Muennighoff, Xi Victoria Lin, Daniela Rus, Bryan Kian Hsiang Low, Sewon Min, Wen tau Yih, Pang Wei Koh, and Luke Zettlemoyer. 2025. ReasonIR: Training retrievers for reasoning tasks. *arXiv:2504.20595*.

Sahel Sharifymoghaddam, Ronak Pradeep, Andre Slavescu, Ryan Nguyen, Andrew Xu, Zijian Chen, Yilin Zhang, Yidi Chen, Jasper Xian, and Jimmy Lin. 2025. RankLLM: A python package for reranking with LLMs. In *Proceedings of the 48th International ACM SIGIR Conference on Research and*

Development in Information Retrieval, SIGIR '25, page 3681–3690, New York, NY, USA. Association for Computing Machinery.

Hongjin SU, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han yu Wang, Liu Haisu, Quan Shi, Zachary S Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan O Arik, Danqi Chen, and Tao Yu. 2025. BRIGHT: A realistic and challenging benchmark for reasoning-intensive retrieval. In *The Thirteenth International Conference on Learning Representations*.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937, Singapore. Association for Computational Linguistics.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Ellen M. Voorhees. 2002. The philosophy of information retrieval evaluation. In *Evaluation of Cross-Language Information Retrieval Systems*, pages 355–370, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ellen M. Voorhees. 2019. *The Evolution of Cranfield*, pages 45–69. Springer International Publishing, Cham.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. Improving text embeddings with large language models. *arXiv:2401.00368*.

Shuai Wang, Ekaterina Khramtsova, Shengyao Zhuang, and Guido Zuccon. 2024b. Feb4RAG: Evaluating federated search in the context of retrieval augmented generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 763–773, New York, NY, USA. Association for Computing Machinery.

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:2504.12516*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *arXiv:2505.09388*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. *arXiv:2506.05176*.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, Yuxin Gu, Sixin Hong, Jing Ren, Jian Chen, Chao Liu, and Yining Hua. 2025. BrowseComp-ZH: Benchmarking web browsing ability of large language models in Chinese. *arXiv:2504.19314*.

Shengyao Zhuang and Guido Zuccon. 2022. Asyncval: A toolkit for asynchronously validating dense retriever checkpoints during training. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 3235–3239, New York, NY, USA. Association for Computing Machinery.

A Problematic Cases

- BrowseComp Errors: During the verification process, we discover that some question-answer pairs in BrowseComp are inherently flawed. For example, one question asks for the name of a book whose author later returned to acting. Using the ground-truth answer, we can identify the intended book and its listed author. However, upon further investigation, we find that the individual who wrote the book and the one who returned to acting are two different people who happen to share the same name.
- Extensive Use of Google Maps: 42 queries in BrowseComp require distance-related information that explicitly prompt multiple calls to Google Maps. These are removed because high-quality documents discussing specific Google Maps distances between arbitrary locations are difficult to obtain. Moreover, scraping static snapshots of Google Maps pages to include in the corpus is not a valid substitute; answering such questions as intended should require agents to be augmented with access to the Google Maps API, rather than retrieving from a corpus. However, this capability lies outside the scope of our objective to build a static, document-based dataset.
- Ambiguous or Non-Unique Answers: Some question-answer pairs are well-supported by documents, but suffer from ambiguity in the expected answer format or the existence of multiple valid answers. For instance, one question asks for the username of an individual who authored a specific story on an internet forum. While the ground-truth answer is correct, it is only one of three usernames credited as authors. We remove 13 such queries due to this kind of ambiguity.

B Scaling Reasoning Effort

We evaluate how the reasoning effort of LLMs influences answer quality and retrieval behavior. To isolate this effect, we focus on the gpt-oss family, which offers three reasoning modes: *low*, *medium*, and *high*. As shown in Table 4, increasing the reasoning effort leads to substantial improvements in accuracy and recall across all model sizes and retrievers. For example, gpt-oss-20B with Qwen3-Embed-8B improves from 13.37% accuracy in *low* mode to 34.58% in *high* mode, along with a recall jump from 17.37% to 49.29%. Similarly, gpt-oss-120B with Qwen3-Embed-8B's accuracy rises from 24.94% to 42.89%. These gains, however, come with a tradeoff: higher reasoning modes dramatically increase the average number of search calls (e.g., from $\approx$2 to $\approx$24 for gpt-oss-20B with Qwen3-Embed-8B), implying higher computational and latency costs. Interestingly, calibration error tends to decrease with higher reasoning effort, suggesting that the models give better confidence estimates as they reason more extensively.

C Effect of Reranking

To evaluate the impact of reranking, we apply listwise reranking (Sun et al., 2023; Ma et al., 2023) over the top-20 and top-100 retrieved candidates using RankLLM (Sharifymoghaddam et al., 2025), with Qwen3-8B/32B and ReasonRank-7B/32B (Liu et al., 2025) models. The reranker uses a sliding window of 20 candidates and a stride of 10, using a 16k-token context and a 16k-token thinking budget (output token count) to balance coverage and compute. Longer candidates are truncated to fit within the context window as needed.

Table 5 reports the effect of reranking after first-stage retrieval with Qwen3-Embed-8B, in the retrieval-only setting. For like-sized models, Qwen3 and ReasonRank perform similarly, with differences typically within 1 point. Overall, reranking yields sizable gains, improving Recall@5 by 8.4–24.0 points. With top-20 reranking, model size matters little (only $\sim$2–3 points difference). Expanding the reranking candidate set to 100 improves all models, with larger gains for the 32B models, widening the effectiveness gap between 8B and 32B models at higher rerank depths.

Table 6 reports the effect of integrating reranking into end-to-end performance of two search agents, GPT-4.1 and gpt-oss-20B (high reasoning effort), using Qwen3-Embed-8B as the first-stage retriever and Qwen3-8B to rerank the top 20 candidates. For both models, Accuracy and Recall improve substantially. This further indicates that reranking improves the precision and recall of retrieved evidence at higher ranks, helping the agent surface more relevant information.

Table 4: OpenAI gpt-oss models in different reasoning effort settings

LLM	Retriever	Accuracy	Recall	Search Calls	Calibration Error
gpt-oss-20B-low	BM25	4.11%	5.36%	1.89	40.89%
	Qwen3-Embed-8B	13.37%	17.37%	1.87	36.34%
gpt-oss-20B-medium	BM25	16.39%	21.96%	13.72	41.78%
	Qwen3-Embed-8B	29.88%	41.31%	13.64	35.99%
gpt-oss-20B-high	BM25	21.08%	31.98%	26.87	33.42%
	Qwen3-Embed-8B	34.58%	49.29%	23.87	27.81%
gpt-oss-120B-low	BM25	9.52%	8.54%	2.06	43.59%
	Qwen3-Embed-8B	24.94%	22.50%	2.21	40.96%
gpt-oss-120B-medium	BM25	23.73%	27.02%	9.73	45.78%
	Qwen3-Embed-8B	37.59%	43.45%	9.64	41.77%
gpt-oss-120B-high	BM25	28.67%	35.50%	19.45	46.48%
	Qwen3-Embed-8B	42.89%	52.63%	18.35	40.34%

Table 5: Effectiveness of rerankers with Qwen3-Embed-8B in retriever-only evaluation. The full question is used as the query in both stages. Reranking is applied to the top-20 and top-100 candidates. Scores in parentheses denote improvements over the base retriever (Δ vs. first stage).

Reranker	Top–20		Top–100	
	Recall@5 (Δ)	nDCG@10 (Δ)	Recall@5 (Δ)	nDCG@10 (Δ)
Qwen3-Embed-8B	14.5 (−)	20.3 (−)	14.5 (−)	20.3 (−)
Evidence Document Retrieval				
ReasonRank-7B	22.9 (+8.4)	29.5 (+9.2)	29.5 (+15.0)	38.0 (+17.7)
Qwen3-8B	23.3 (+8.8)	30.0 (+9.7)	29.6 (+15.1)	37.7 (+17.4)
ReasonRank-32B	24.9 (+10.4)	32.1 (+11.8)	34.4 (+19.9)	43.8 (+23.5)
Qwen3-32B	24.7 (+10.2)	31.8 (+11.5)	35.0 (+20.5)	44.3 (+24.0)
Gold Document Retrieval				
ReasonRank-7B	28.7 (+10.2)	28.9 (+9.4)	36.8 (+18.3)	37.1 (+17.6)
Qwen3-8B	29.2 (+10.7)	29.6 (+10.1)	36.7 (+18.2)	36.6 (+17.1)
ReasonRank-32B	30.7 (+12.2)	31.5 (+12.0)	42.5 (+24.0)	43.5 (+24.0)
Qwen3-32B	30.5 (+12.0)	31.3 (+11.8)	42.2 (+23.7)	43.0 (+23.5)

Table 6: Effect of reranking on end-to-end agent performance. Qwen3-Embed-8B is used as the first-stage retriever and Qwen3-8B is used for reranking top 20 retrieved candidates.

LLM	Retriever/Reranker	Accuracy	Recall	Search Calls	Calibration Error
GPT-4.1	Qwen3-Embed-8B	35.42%	36.89%	8.67	54.67%
	+Qwen3-8B	47.11%	51.46%	8.77	49.86%
gpt-oss-20B-high	Qwen3-Embed-8B	34.58%	49.29%	23.87	27.81%
	+Qwen3-8B	40.24%	57.98%	21.98	21.47%

D Effect of Corpus Size

The corpus in `BrowseComp-Plus` contains approximately 100K documents. While real-world agents often operate over much larger, web-scale corpora, we aim to assess whether our designed corpus size is sufficient to support valid experimental observations. To this end, we augment our benchmark corpus with the FineWeb-edu (Penedo et al., 2024) document collection (10 billion tokens),[3] deduplicated by URL. This expansion results in a significantly larger corpus of 9,771,311 documents.

Table 7 shows retrieval effectiveness before and after adding FineWeb documents. For BM25, retrieval effectiveness improves across all metrics, likely due to better inverse document frequency (IDF) estimation in the larger corpus, which strengthens BM25's lexical scoring.

In contrast, neural retrievers (Qwen3-Embedding-8B and ReasonIR-8B) show degraded effectiveness on the FineWeb-augmented corpus. This drop is theoretically expected: the relative ranking of documents from the original small corpus remains unchanged, but the newly added FineWeb documents can now appear in the top ranks. Since these additional documents are unjudged, they are treated as non-relevant under standard TREC-style evaluation, inevitably lowering measured retrieval effectiveness.

It is important to note that lower retrieval scores for embedding models on FineWeb do not necessarily indicate worse final answers; some unjudged, top-ranked FineWeb documents may be "false negatives" that still provide useful evidence. However, as shown in Table 8, adding FineWeb does not improve end-to-end accuracy for embedding-based retrievers. For instance, Qwen3-32B with Qwen3-Embedding-8B drops from 10.36% to 7.11%.

Overall, substantially expanding the corpus size does not lead to different conclusions about the ranking or effectiveness level among the retrievers and agents, supporting our claim that the original 100K corpus offers both strong positive coverage and sufficient challenge for robust evaluation.

E Details of Annotation

We recruited 14 university students in an information retrieval research group, 6 of whom are current or completed PhD students specializing in information retrieval. Each annotator underwent approximately 1 hour of training on the labeling task on an internal development set prior to the real annotation on `BrowseComp-Plus`. Additionally, besides detailed text instructions, we created a 40-minute long demonstration video, covering many edge cases, which the annotators could constantly refer to. Further unsure cases were consolidated in a group channel, possibly relabeling prior cases for consistency. After the labeling process, 10 examples from each labeler were randomly sampled and discussed in the group channel, showing over 80% agreement. For the cases where a labeler made a mistake, the labeler was instructed to relabel all of their prior examples, avoiding similar mistakes.

[3] https://huggingface.co/datasets/HuggingFaceFW/fineweb-edu/viewer/sample-10BT

F Significance Test of Main Results

In Figure 11, we present the visualization of the significance test on the answer accuracy of each search agent integrated with different retrievers. The methods are ordered by their accuracy scores. Pairwise McNemar's tests at $p \leq 0.05$ were conducted, where a green cell at Row (i), Column (j) indicates that the method in Row (i) performs significantly better than the method in Column (j).

G Answer Accuracy with Different Judgment Methods

In Table 10, we report answer-accuracy measurements using LLM-as-judge with GPT-4.1, Qwen3-32B, and substring match (whether the answer was included as a substring in the response). We observe that the various evaluation methods are consistent. Notably, upon human inspection, we find that the LLM-as-judge approach is more robust in handling cases where the predicted answers differ in format from the ground-truth labels.

H Future Work and Discussion

We believe that our `BrowseComp-Plus` opens new avenues for advancing research in the deep search area. `BrowseComp-Plus` retains the challenging nature of the original BrowseComp while providing a more controlled and transparent experimental setup similar to early pivotal evaluation benchmarks like Natural Questions (NQ) (Kwiatkowski et al., 2019) and HotpotQA (Yang et al., 2018). Like how NQ and HotpotQA have facilitated the design, comparison, and development of modern neural QA systems, we hope that `BrowseComp-Plus` will serve similar roles for deep search agent studies. Here, we list some immediate research directions.

Table 7: Evidence document retrieval effectiveness on the FineWeb 10BT corpus.

Retriever	Corpus	Recall@5	Recall@100	Recall@1000	nDCG@10
BM25	Original	1.2	4.7	13.6	1.6
BM25	Original + FineWeb	2.2	8.0	19.4	3.1
Qwen3-Embed-8B	Original	14.5	47.7	76.7	20.3
Qwen3-Embed-8B	Original + FineWeb	11.6	37.6	64.2	16.4
ReasonIR-8B	Original	12.2	43.6	73.9	16.8
ReasonIR-8B	Original + FineWeb	8.6	30.7	56.3	11.8

Table 8: Accuracy of end-to-end search agents on our `BrowseComp-Plus` original 100k corpus vs. FineWeb 10BT corpus.

LLM	Retriever	Corpus	Accuracy
SearchR1-32B	BM25	Original	3.86%
	BM25	Original + FineWeb	4.72%
	Qwen3-Embed-8B	Original	10.36%
	Qwen3-Embed-8B	Original + FineWeb	8.33%
Qwen3-32B	BM25	Original	3.49%
	BM25	Original + FineWeb	5.42%
	Qwen3-Embed-8B	Original	10.36%
	Qwen3-Embed-8B	Original + FineWeb	7.11%

While our current work focuses on how different retrievers influence inference performance, a promising future direction is to examine the role of the retriever during agent optimization. For example, optimizing a search agent may be more challenging when paired with BM25 than with a modern embedding-based retriever, simply because BM25 surfaces fewer relevant documents. Understanding how retriever quality affects the learning dynamics of an agent remains an open question.

Another important extension is to study the agent's out-of-distribution tool-use capabilities. For instance, if an agent is optimized using a BM25 search tool, how well does it generalize when switched to an embedding-based search tool?

A more creative research direction would be to attempt a breakdown of the commercial search engine. As much as a folktale, a commercial search solution employs tiered, composed, and multi-faceted retrieval pipelines. Is the LLM able to orchestrate a set of search tools to perform federated search (Wang et al., 2024b), or even a sub-agent, to get quality results similar to those from Google?

A further direction is to design retrieval models that are tolerant of, or even adaptive to, a specific agent. In the deep search setting, the primary consumer of retrieved documents is no longer a human, but a tool-augmented LLM agent. This raises the possibility that retrieval models could be co-optimized with the agent for achieving overall answer accuracy, rather than developed and evaluated in isolation.

Overall, `BrowseComp-Plus` serves as an ideal testbed for pursuing these directions, enabling systematic and fine-grained analyses of agent-retriever interactions within deep search systems.

I Usage of LLM

ChatGPT is used during the writing to polish text (e.g., correct grammar) and format tables.

Table 9: Overall API costs of proprietary agents for the experiments in Table 1.

LLM	Retriever	Accuracy	Price (USD)
GPT-4.1	BM25	14.58%	$106.96
	Qwen3-Embed-8B	35.42%	$89.81
o3	BM25	49.28%	$836.35
	Qwen3-Embed-8B	63.49%	$740.79
GPT-5	BM25	55.90%	$400.36
	Qwen3-Embed-8B	70.12%	$360.71
Sonnet 4	BM25	14.34%	$352.04
	Qwen3-Embed-8B	36.75%	$325.75
Opus 4	BM25	15.54%	$2,043.95
	Qwen3-Embed-8B	36.14%	$1,842.48
Gemini 2.5 Flash	BM25	15.54%	$47.32
	Qwen3-Embed-8B	33.01%	$41.29
Gemini 2.5 Pro	BM25	19.04%	$138.64
	Qwen3-Embed-8B	28.67%	$99.92

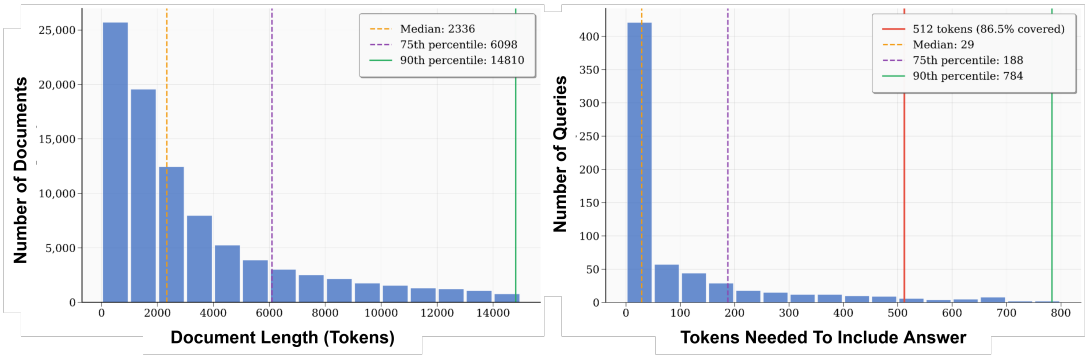

Figure 3: (a) Token distribution of corpus document length, showing up to 90th percentile for display; (b) Distribution of tokens needed to include answer in gold documents per query, showing up to 90th percentile for display

I will give you a question and a correct answer, and you are to search online for evidence that supports the answer. List the evidence you've used to justify this answer step-by-step, including their urls in your output. Your final list of urls should be in the order such that a human can visit them in order to justify the answer.

Question: {question}

Answer: {answer}

This is all the information you have to work with to produce the final list of urls. Format your answer in a table with 3 columns:
- clue: the clue mentioned in the question
- url: the http web url of the evidence you've found
- evidence: the content in the url page that supports the clue

Figure 4: Prompt used for OpenAI o3 evidence document gathering.

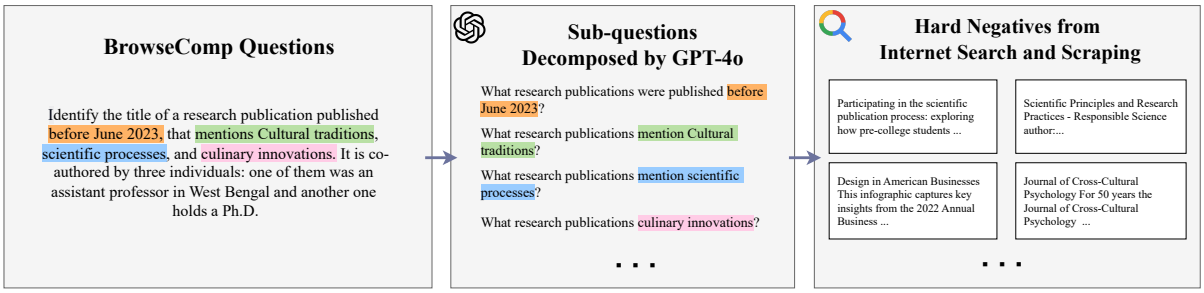

Figure 5: The pipeline of collecting hard negative documents in Section 3.3.

You are an expert at breaking down complex, multi-part questions into simpler, self-contained subqueries.

Your task is to analyze the given question and decompose it into a series of smaller, more manageable subqueries that, when answered together, would provide all the information needed to answer the original question.

Guidelines:

1. Each subquery should focus on a single piece of information or concept
2. Subqueries MUST be completely self-contained and answerable independently - do not use pronouns or references like "this person", "the author", "these conditions", "they", "the movie", etc.
3. Each subquery should include all necessary context and constraints from the original query
4. Preserve all important details and constraints from the original query
5. Return only the subqueries as a JSON array of strings

Example:

Original: "Please identify the fictional character who occasionally breaks the fourth wall with the audience, has a backstory involving help from selfless ascetics, is known for his humor, and had a TV show that aired between the 1960s and 1980s with fewer than 50 episodes."

Subqueries: ["Which fictional characters occasionally break the fourth wall with the audience?", "Which fictional characters have a backstory involving help from selfless ascetics?", "Which fictional characters are known for their humor?", "Which TV shows aired between the 1960s and 1980s?", "Which TV shows had fewer than 50 episodes?]

Please decompose this query into subqueries:
{query}

Figure 6: Prompt used to decompose queries during negative mining.

You are a deep research agent. You need to answer the given question by interacting with a search engine, using the search tool provided. Please perform reasoning and use the tool step by step, in an interleaved manner. You may use the search tool multiple times.

Question: {Question}

Your response should be in the following format:
Explanation: {{your explanation for your final answer. For this explanation section only, you should cite your evidence documents inline by enclosing their docids in square brackets [] at the end of sentences. For example, [20].}}
Exact Answer: {{your succinct, final answer}}
Confidence: {{your confidence score between 0% and 100% for your answer}}

Figure 7: The main prompt given to agents.

Judge whether the following [response] to [question] is correct or not based on the precise and unambiguous [correct_answer] below.

[question]: {question}

[response]: {response}

Your judgement must be in the format and criteria specified below:

extracted_final_answer: The final exact answer extracted from the [response]. Put the extracted answer as 'None' if there is no exact, final answer to extract from the response.

[correct_answer]: {correct_answer}

reasoning: Explain why the extracted_final_answer is correct or incorrect based on [correct_answer], focusing only on if there are meaningful differences between [correct_answer] and the extracted_final_answer. Do not comment on any background to the problem, do not attempt to solve the problem, do not argue for any answer different than [correct_answer], focus only on whether the answers match.

correct: Answer 'yes' if extracted_final_answer matches the [correct_answer] given above, or is within a small margin of error for numerical problems. Answer 'no' otherwise, i.e. if there is any inconsistency, ambiguity, non-equivalency, or if the extracted answer is incorrect.

confidence: The extracted confidence score between 0% and 100% from [response]. Put 100 if there is no confidence score available.

Figure 8: Prompt used to perform evaluation.

You are a deep research agent. You need to answer the given question by interacting with a search engine, using the search and get_document tools provided. Please perform reasoning and use the tools step by step, in an interleaved manner. You may use the search and get_document tools multiple times.

Question: {Question}

Your response should be in the following format:

Explanation: {{your explanation for your final answer. For this explanation section only, you should cite your evidence documents inline by enclosing their docids in square brackets [] at the end of sentences. For example, [20].}}
Exact Answer: {{your succinct, final answer}}
Confidence: {{your confidence score between 0% and 100% for your answer}}

Figure 9: Modified search prompt to also use the get-document tool.

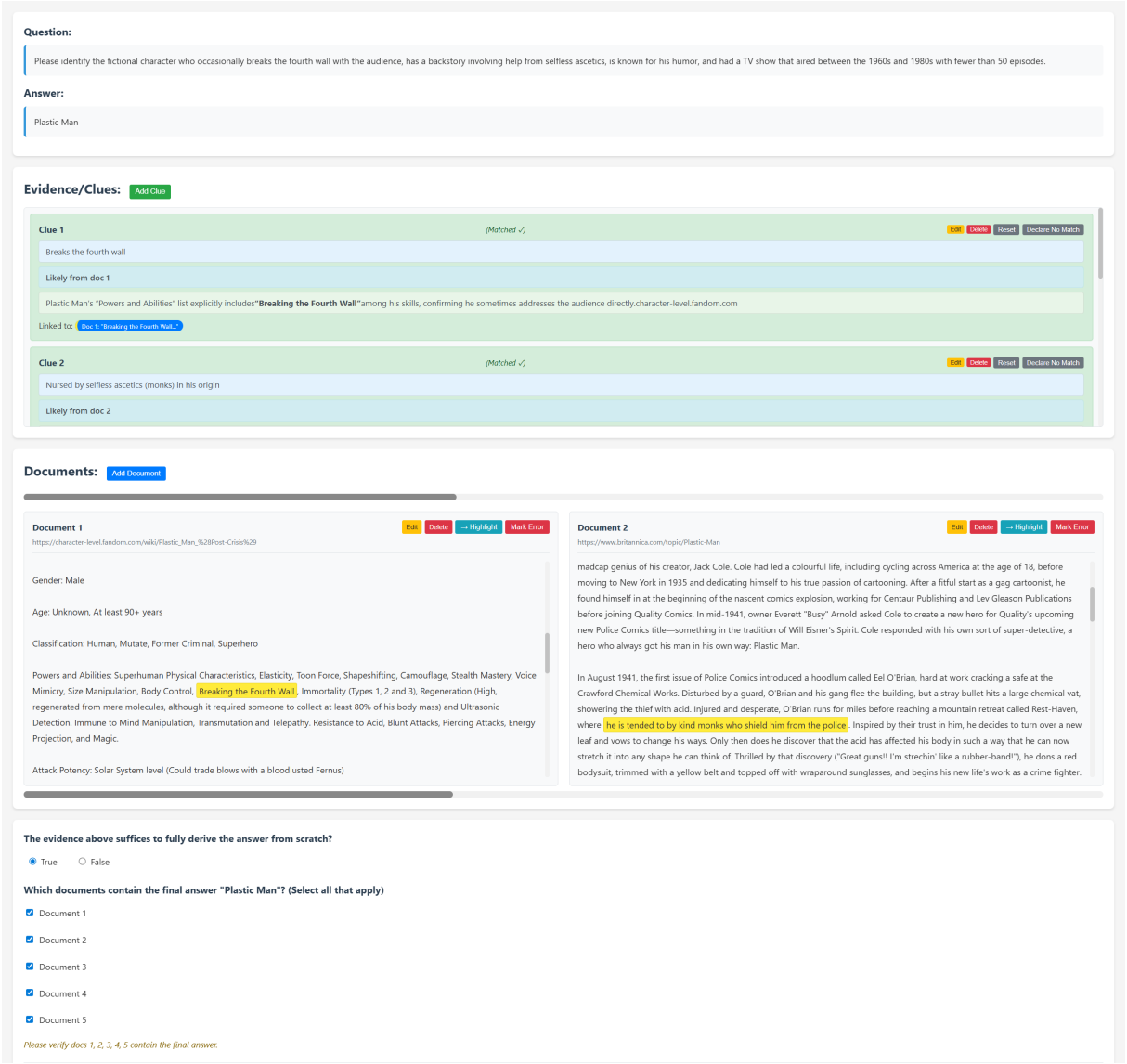

Figure 10: A screenshot example of the annotation interface.

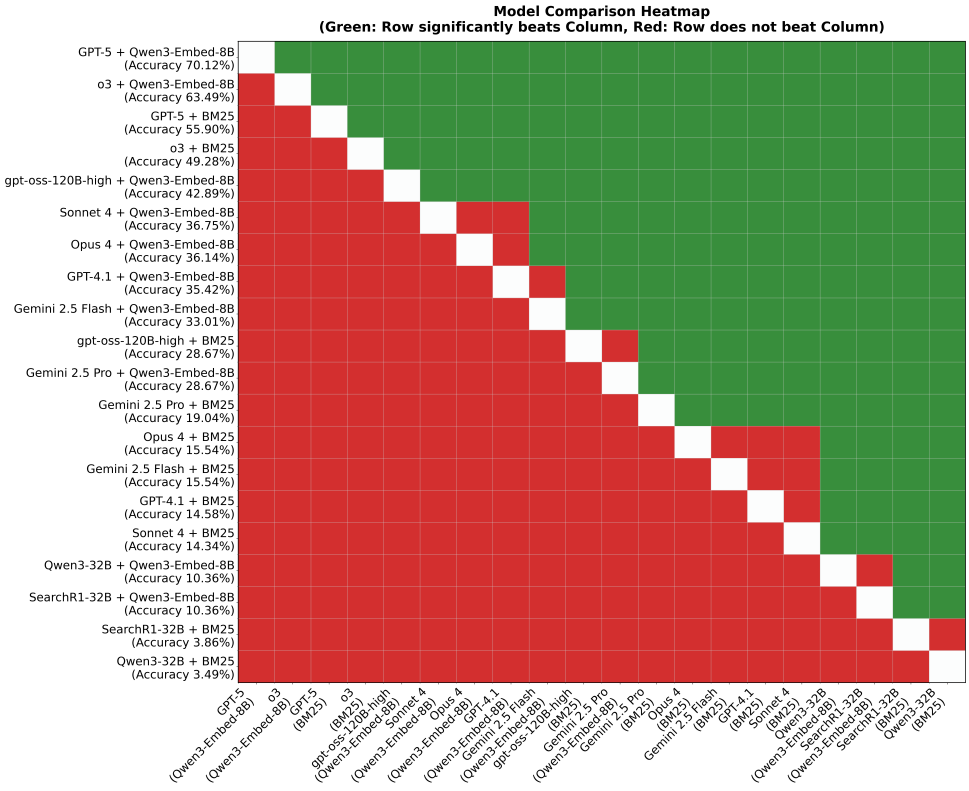

Figure 11: Pairwise McNemar's tests at a significance level of $p \leq 0.05$. A green cell at Row (i), Column (j) indicates that the method in Row (i) performs significantly better than the method in Column (j).

LLM	Retriever	Substring Match	GPT-4.1 Judge	Qwen3-32B Judge
GPT-4.1	BM25	14.58	14.58	15.30
GPT-4.1	Qwen3-Embedding-8B	34.46	35.42	36.39
o3	BM25	45.78	49.28	50.48
o3	Qwen3-Embedding-8B	60.48	63.49	65.90
Sonnet 4	BM25	13.37	14.34	14.70
Sonnet 4	Qwen3-Embedding-8B	33.73	36.75	37.35
Opus 4	BM25	15.18	15.54	15.54
Opus 4	Qwen3-Embedding-8B	33.13	36.14	36.75
Gemini 2.5 Flash	BM25	15.54	15.54	16.27
Gemini 2.5 Flash	Qwen3-Embedding-8B	31.45	33.01	34.58
Gemini 2.5 Pro	BM25	17.71	19.04	19.88
Gemini 2.5 Pro	Qwen3-Embedding-8B	27.83	28.67	29.52
Qwen3-32B	BM25	3.25	3.49	3.61
Qwen3-32B	Qwen3-Embedding-0.6B	4.22	4.10	4.22
Qwen3-32B	Qwen3-Embedding-4B	8.43	7.83	8.07
Qwen3-32B	Qwen3-Embedding-8B	9.76	10.36	10.72
Qwen3-32B	ReasonIR	8.67	9.16	9.28
SearchR1-32B	BM25	3.86	3.86	4.11
SearchR1-32B	Qwen3-Embedding-0.6B	6.27	5.66	6.02
SearchR1-32B	Qwen3-Embedding-4B	10.60	9.40	9.28
SearchR1-32B	Qwen3-Embedding-8B	11.81	10.36	11.08
SearchR1-32B	ReasonIR	10.64	9.43	9.31
gpt-oss-20B-low	BM25	3.51	4.11	3.99
gpt-oss-20B-low	Qwen3-Embedding-8B	11.93	13.37	14.10
gpt-oss-20B-medium	BM25	15.54	16.39	16.87
gpt-oss-20B-medium	Qwen3-Embedding-8B	26.87	29.88	30.48
gpt-oss-20B-high	BM25	19.76	21.08	21.45
gpt-oss-20B-high	Qwen3-Embedding-8B	31.93	34.58	35.06
gpt-oss-120B-low	BM25	8.80	9.52	9.76
gpt-oss-120B-low	Qwen3-Embedding-8B	22.41	24.94	25.54
gpt-oss-120B-medium	BM25	21.33	23.73	24.58
gpt-oss-120B-medium	Qwen3-Embedding-8B	33.49	37.59	38.55
gpt-oss-120B-high	BM25	26.99	28.67	29.16
gpt-oss-120B-high	Qwen3-Embedding-8B	40.24	42.89	44.10
GPT-5	BM25	51.69	55.90	57.59
GPT-5	Qwen3-Embedding-8B	65.18	70.12	71.69

Table 10: Comparison of accuracy measurement based on LLM-as-judge with GPT-4.1, LLM-as-judge with Qwen3-32B, and substring match.