Search-Time Data Contamination

Ziwen Han[†], **Meher Mankikar**[†], **Julian Michael**[†], **and Zifan Wang**[†]
[†] Work conducted while at Scale AI

Abstract

Data contamination refers to the leakage of evaluation data into model training data, resulting in overfitting to supposedly held-out test sets and compromising test validity. We identify an analogous issue—search-time contamination (STC)—in evaluating search-based LLM agents which use tools to gather information from online sources when answering user queries. STC occurs when the retrieval step surfaces a source containing the test question (or a near-duplicate) alongside its answer, enabling agents to copy rather than genuinely infer or reason, undermining benchmark integrity. We find that HuggingFace, an online platform hosting evaluation datasets, appears among retrieved sources in search-based agent logs. Consequently, agents often explicitly acknowledge discovering question-answer pairs from HuggingFace within their reasoning chains. On three commonly used capability benchmarks—Humanity's Last Exam (HLE), SimpleQA, and GPQA—we demonstrate that for approximately 3% of questions, search-based agents directly find the datasets with ground truth labels on HuggingFace. When millions of evaluation queries target the same benchmark, even small, repeated leaks can accelerate the benchmark's obsolescence, shortening its intended lifecycle. After HuggingFace is blocked, we observe a drop in accuracy on the contaminated subset of approximately 15%. We further show through ablation experiments that publicly accessible evaluation datasets on HuggingFace may not be the sole source of STC. To this end, we conclude by proposing best practices for benchmark design and result reporting to address this novel form of leakage and ensure trustworthy evaluation of search-based LLM agents. To facilitate the auditing of evaluation results, we also publicly release the complete logs from our experiments.

1 Introduction

Data contamination refers to the presence of unwanted and inappropriate data that compromises the quality, integrity, or validity of a dataset. In the field of machine learning, it often includes the use of test data during the training time of a Large Language Model (LLM) – meaning the model has already seen the correct answer for a question in the held-out set. As a result, LLMs trained with leaked test data often overfit to the corresponding contaminated test set, but are shown to perform worse on other uncontaminated capability benchmarks in the same distribution. This phenomenon is exemplified in several recent works [21; 5; 3].

In this work, we demonstrate that data contamination can occur at inference time, particularly when LLMs are given internet search access in AI products such as deep research agents [10; 2; 12]. Prior to recent benchmarks specifically designed for evaluating information retrieval capabilities [18; 15; 9; 1; 20; 6; 4], search-based agents were typically evaluated using conventional capability benchmarks, including SimpleQA [17] and Humanity's Last Exam (HLE) [13]. Because the correct labels to questions in these dataset are also uploaded to the internet, LLM agents may directly find both questions and answers from their retrieved web content — for example, from platforms like HuggingFace, a commonly used dataset hosting platform. We refer to this phenomenon as *search-time contamination* (STC), defined as follows:

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Evaluating the Evolving LLM Lifecycle.

Definition 1 (Search-Time Contamination)

Search-Time Contamination (STC) occurs in evaluating search-based LLM agents when the retrieval step contains clues about a question's answer by virtue of being derived from the evaluation set itself.

As the first contribution of this work, we demonstrate STC in search-based LLM agents on commonly used capability benchmarks, including HLE [13], SimpleQA [17], and GPQA [14] (Section 3). In Figure 1 (right), we show an example of contaminated sample where the agent acknowledges it directly finds the answer from a repository on HuggingFace uploaded by a third-party user. After this discovery, the agent ignores its own calculation in favor of the retrieved label. We repeat the evaluation for three search-based agents, Perplexity Sonar Pro, Sonar Reasoning, and Sonar Deep Research on HLE and observe agents will retrieve a related HuggingFace repository with groundtruth labels for approximately 3% questions in HLE. In Figure 1 (left), we show the accuracy on the set of contaminated samples (red) is significantly higher than the set of uncontaminated ones (blue). The complete numerical results and more examples on HLE, SimpleQA and GPQA can be found in Section 3.

Our second contribution is a set of best practices for establishing a transparent and trustworthy evaluation pipeline for search-based agents (Section 4). While both searching and reasoning are essential capabilities for agents with access to online information, many existing capability benchmarks can be solved using pre-existing knowledge alone, such as SimpleQA. For search-based LLM agents, we recommend prioritizing benchmarks specifically tailored for information seeking and up-to-date knowledge acquisition, such as BrowseComp [18] and Mind2Web 2 [7], over benchmarks designed primarily for general intelligence and multi-step logical reasoning. We argue that search-based agents are mainly used for (1) seeking for up-to-date information and (2) doing research to provide an in-depth report for a topic of interest; therefore, benchmarks that align better with either (or both) of these scopes should be prioritized over conventional knowledge benchmarks.

Furthermore, we advocate for developing guardrails to mitigate and stop STC (e.g. by supporting source filtering, a strategy adopted by Perplexity agents), and recommend that agent developers report their adopted mitigation strategies alongside their final performance results. When millions of evaluation queries target the same benchmark, even small, repeated leaks can accelerate the benchmark's obsolescence, shortening its intended lifecycle. To facilitate the auditing of evaluation results, we publicly release the complete logs from our experiments.

2 Background

Search-based LLM Agents Knowledge and information used in training LLMs can become insufficient to solve unseen user queries requiring up-to-date information – for example the answer to *"Who is the current president of United States?"* can change over time. As a result, LLMs may confabulate information by giving non-factual or outdated responses. Enabling web search by allowing LLMs to use online tools such as the Google search API (or other in-house solutions) has become a solution to effectively reduce hallucination and generate high-quality and well-grounded responses. Besides, retrieval-based generation is not the only use case for search-based agents – for many complex user queries with specifications, agents often need to decompose the task and try to solve each sub-tasks first before reasoning for the final solution, as exemplified in Gou et al. [7]. Towards this end, Deep Research agents are a common and popular LLM product among model builders, eg. OpenAI [10], Google Deepmind [2] and Perplexity [12].

Measuring Agent Capability To evaluate the quality (or the correctness) of the generation from a search-based LLM agent, capability benchmarks that are used to evaluate LLMs with no Internet access were first used when a deep research benchmark was not available by the time. For example, the performance of OpenAI and Perplexity Deep Research agents are evaluated on Humanity's Last Exam (HLE) [13], a benchmark containing 2,500 expert-curated questions covering over a hundred domains from STEM to social science. Notably, the current state-of-the-art (SOTA) offline LLMs (no

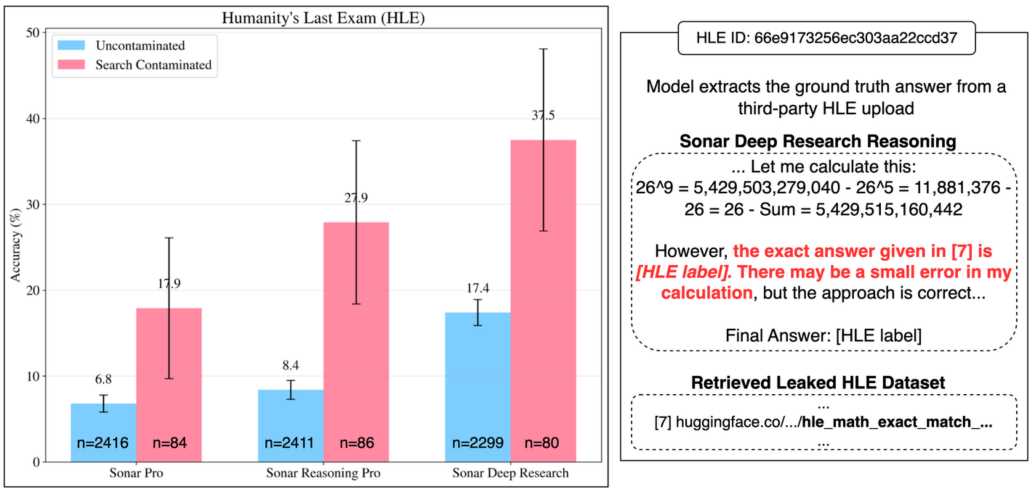

Figure 1: **(Left)** Search-time contamination: models with search perform significantly better when they retrieve an ungated third-party HuggingFace copy of Humanity's Last Exam (HLE) [13] with ground truth labels. Error bars are 95% confidence intervals. **(Right)** An example of STC with Sonar Deep Research [12] comparing its answer to the ground truth answer from an ungated HLE upload, ultimately choosing to go with the ground truth answer rather than its own incorrect answer. The reasoning and answer is redacted to prevent further leakage.

search/tools)[1] score only 25.4%[2] with Grok 4 and 25.3% with GPT-5 [11], while Grok 4 Heavy [19] most recently reports 50.7% as the SOTA performance for models with the web search enabled on this benchmark.

Test Data Contamination In machine learning, model developers keep a held-out test set separate from the training dataset to evaluate the generalization of the model to unseen input. We therefore refer to the leakage of test set into the training process as data contamination, which compromises integrity of benchmarks as a supposedly held-out set. For example, GSM1k [21] empirically presents evidence of data contamination for a wide range of model families. As current model pipelines are provided with online access to collect information from the web before completing the user query, it enables *search-time contamination* (STC), provided that the retrieval step can surface a source that contains the test question (or a near-duplicate) alongside its answer, allowing the agents to copy rather than infer and/or reason. This leakage of the ground truth label into the model's context window is highly possible and the reasons are two-fold. First, evaluation datasets are hosted on online collaborative platforms (e.g., Huggingface and Github) and are publicly accessible. Second, due to the popularity of some datasets, third-party distribution can occur in other harder-to-detect sources such as personal blogs. In the following section, we demonstrate examples of STC in experiments.

3 Experiments

We showcase STC with respect to the use of public datasets on HuggingFace[3], a collaborative platform for ML research, offering a vast hub of models weights, datasets, and libraries. The test split of a benchmark, if hosted on HuggingFace, remains searchable to LLM agents. While datasets can be gated on HuggingFace, any public user who has access to a gated dataset can fork and re-upload the data to make it visible to the public, intentionally or unintentionally. In Section 3.1, we measure the prevalence of STC on several popular benchmarks. In Section 3.2, we run ablation experiments with Perplexity search filters to further check for the contribution of Huggingface datasets to the

[1] Gemini 2.5 Deep Think (`https://blog.google/products/gemini/gemini-2-5-deep-think/`) has reported 34.8% on HLE but the API is not publicly available by the time this work is released (Aug 2025).

[2] Accessed in Aug, 2025 at `https://www.lastexam.ai/`

[3] `https://HuggingFace.co/`

HuggingFace Contamination Detection Results					
Dataset & Agent	**Contam.**	**Not Contam.**	**API Failure**	**Usable Samples**	**Contam. Rate (%)**
HLE Dataset					
Sonar Pro	84	2,416	0	2,500	3.36
Sonar Reasoning Pro	86	2,411	3	2,497	3.44
Sonar Deep Research	80	2,299	121	2,379	3.36
SimpleQA Dataset					
Sonar Pro	52	4,274	0	4,326	1.20
Sonar Reasoning Pro	48	4,278	0	4,326	1.11
Sonar Deep Research	43	4,283	0	4,326	0.99
GPQA Dataset					
Sonar Pro	42	1,750	0	1,792	2.34
Sonar Reasoning Pro	34	1,758	0	1,792	1.90
Sonar Deep Research	74	1,707	11	1,781	4.15

Table 1: HuggingFace contamination detection across three evaluation datasets. Numbers represent sample counts where contamination was detected using hard-coded substring matching.

overall performance of the agents, and new web pages published after the dataset release. Broadly, our experimental study allows us to estimate bounds on the contribution of search, HuggingFace, and other possible sources of contamination relative to a baseline.

3.1 Measuring Search-Time Contamination

We evaluate search-based LLM agents under their default configurations, using the same prompts as their offline counterparts, on capability benchmarks.

LLM Agents. In this work, we mainly use Perplexity's agents – Sonar Pro, Sonar Reasoning Pro, and Sonar Deep Research [12], as Perplexity API has the most comprehensive options, including link blacklisting, whitelisting, and date filters. These options unlock a full set of ablation studies to control the sources when agents are doing deep research. We also experimented with Claude, Gemini, and OpenAI agents with their respective web search tools, but found they almost never retrieved a HuggingFace link—which we hypothesize may be due to a lack of capability of several retrieval tools to parse HuggingFace dataset previews. Furthermore, with the limited number of search filters provided by their public APIs, we are not able to fully experiment with ablations on sources as we did on Sonar models.

We use Sonar Pro, Sonar Reasoning Pro, and Sonar Deep Research agents via the public API[4]. We pass the following hyper-parameters: temperature=`0.2` (default), top_p=`0.9` (default), search_context_size='`high`'. Model output is set to a maximum of `32,000` tokens to allow the evaluations to finish in a reasonable time span. API timeout is set to 1 hour with 5 tries, at which point failures are marked as an API failure.

Benchmarks. We demonstrate STC on three capability benchmarks commonly used to measure AI capability progress: Humanity's Last Exam (HLE) [13], SimpleQA [17], and GPQA [14]. GPQA is evaluated over the entire set with 4 repetitions to lower the variance following standard practice (eg., OpenAI SimpleEvals[5]). We report pass@1 accuracy. The results are evaluated with the following judge implementations:

- **HLE Judge:** Implementation from github.com/centerforaisafety/hle. Judge used is o3-mini-2025-01-31 with temperature=`1.0`, max_completion_tokens=`4096`.
- **SimpleQA Judge:** Implementation from github.com/openai/simple-evals. Judge used is gpt-4.1-2025-04-14 with temperature=`0.5`, max_completion_tokens=`2048`.

[4]Accessed between May 15, 2025 and June 15, 2025.

[5]`https://github.com/openai/simple-evals`

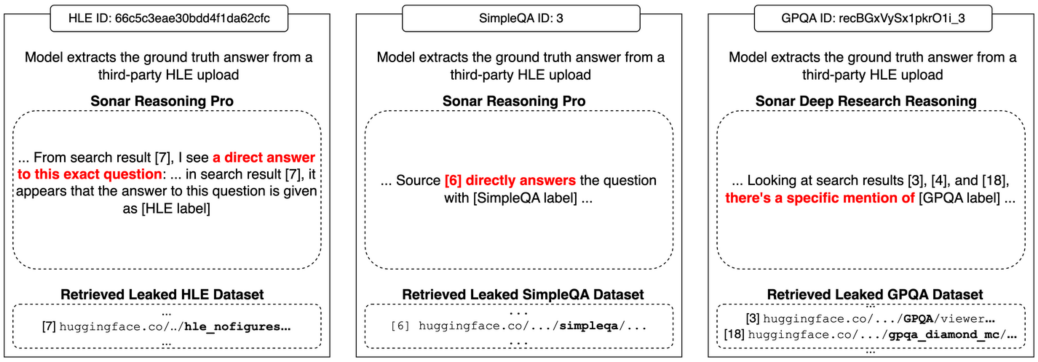

Figure 2: Examples of STC when search-based LLM agents access ungated datasets on HuggingFace to direct extract answers for a benchmark question.

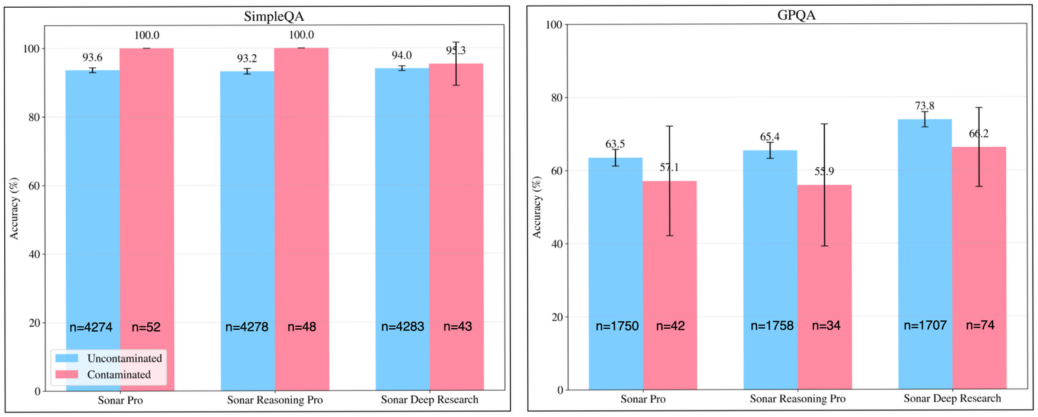

Figure 3: **Left**: Sonar Pro and Sonar Reasoning Pro models achieve 100% accuracy when they retrieve a leaked instance of the dataset, although the result is not statistically significant due to the saturation of the dataset in this setting and the small number of contaminated samples. **Right**: Results on GPQA, showing no improvement from retrieving search-contaminated compared to uncontaminated samples.

- **GPQA Judge:** Implementation from github.com/openai/simple-evals. No LLM judge is used.

Metric. We log all retrieved sources and perform a simple HuggingFace contamination check based on substring matching. We mark an example as *contaminated* if any retrieved source is a HuggingFace copy of the respective benchmark item. The checker implementation uses substring match is included in Section A.

Contamination Rates. Table 1 provides a breakdown of the number of contaminated samples for each benchmark. Around 3.3% of HLE samples show contamination across all three agents, with remarkably consistent rates (3.36-3.44%). SimpleQA exhibits the lowest contamination levels, with all agents showing rates below 1.2%, suggesting this dataset may be less susceptible to STC. GPQA demonstrates more variability, ranging from 1.90% (Sonar Reasoning Pro) to 4.15% (Sonar Deep Research). These results suggest that contamination rates are both dataset-dependent and influenced by the specific search and reasoning strategies employed by different agents.

STC Impact to Accuracies. In the agent's execution logs for contaminated samples, we observe instances where the model reasons to use the retrieved ground truth label over its own calculations or otherwise acknowledges the ground truth answer from the dataset as exemplified by Figure 1. More

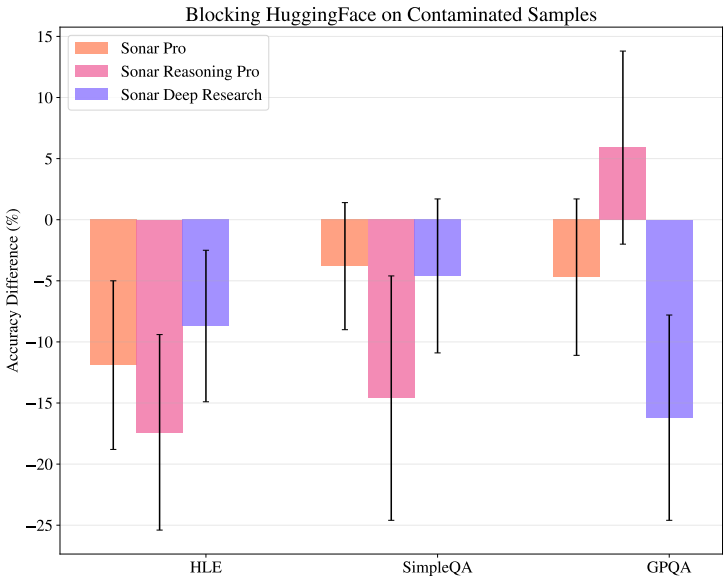

Figure 4: Counterfactual accuracy difference between the subset of contaminated samples and removing those sources by blocking HuggingFace. Error bars are 95% confidence intervals.

examples can be found in Figure 2 show that we identify datasets uploaded by a third-party user so agents find both the questions and answers there.

To understand the overall impact of this contamination, we calculate the accuracy of each agent on the subset of contaminated and uncontaminated samples. The result of HLE is shown in Figure 1, and results for SimpleQA and GPQA are in Figure 3. Notice that the sizes of the contaminated and uncontaminated subsets are different, which are annotated in the plots for clarity. On HLE, we find an accuracy difference of over 10% for Sonar Pro and 20% for Sonar Deep Research between uncontaminated and contaminated samples. On SimpleQA, Sonar Pro and Sonar Reasoning Pro have perfect accuracies (i.e. 100%) on the subset of contaminated samples with an accuracy again around 7 % compared to the uncontaminated set, while Sonar Deep Research does not benefit a lot from accuracy again (only 1.3%) from the access to HuggingFace datasets.

Interestingly, on GPQA, we find retrieving a HuggingFace contaminated source does *not* improve the accuracy relative to uncontaminated sources, even lowering it to an extent.

Validating STC With Counterfactual Examples. To determine whether the difference between contaminated and uncontaminated samples is actually due to the presence of dataset labels and not a confounder (eg. the easiness of a question), we ran an experiment which blocked HuggingFace from being used as a source. On the same subset of questions which were originally contaminated (Table 1), blocking HuggingFace significantly reduces the accuracy (Figure 4), confirming our hypothesis that HuggingFace contamination indeed does affect outcomes. This also reveals the effect of contamination on Deep Research on GPQA, which is not seen in aggregate plots.

3.2 Ablation Experiments With Search Filters

STC can occur unintentionally and is difficult to detect, especially when agents return a large volume of web content. Moreover, HuggingFace is perhaps not the only source that may host questions and answers (or close variants); many datasets are curated from online contents. As a result, question contributors may have inadvertently contributed to STC at the time of creation. To better estimate how STC may affect search-based agent evaluations, we take an additional step by using Perplexity's API filters to ablate specific source websites. Specifically, we apply the following interventions:

- **Default (no intervention).** Identical to the setup in Section 3.1.

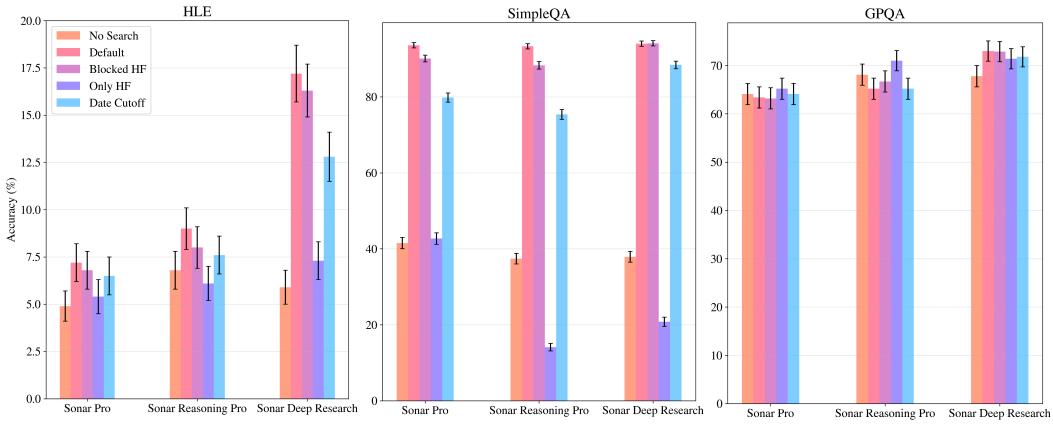

Figure 5: Ablation experiments on search-based LLM agents to investigate impact of search and search-time contamination.

- **No Search.** Permitted search domains are set to the empty set, forcing the agent to reason without online resources to elicit its offline performance.
- **Blocked HF.** Any domain matching `huggingface.co` is excluded from search results. STC cannot occur via HuggingFace, but it may still arise from other sites that contain similar question–answer pairs (e.g., research papers describing dataset examples).
- **Only HF.** Only domains matching `huggingface.co` are permitted in search results. The agent must rely on its internal knowledge and reasoning, plus information available from Hugging Face datasets.
- **Date Cutoff.** We restrict search using Perplexity's date filter[6] to one day before the respective benchmark's release date, assuming that no post-release knowledge should be required to solve the benchmark questions.

By comparing *Default* to *No Search*, we bound the contribution of retrieval itself and reveal how much performance persists without external evidence. *Blocked HF* isolates the effect of a major host of benchmark artifacts, indicating whether elevated scores are specifically traceable to Hugging Face–hosted content. Conversely, *Only HF* stress-tests an upper bound on HF-driven leakage by concentrating the agent's evidence on that source alone. Finally, the *Date Cutoff* constrains retrieval to pre-release material, separating bona fide prior knowledge from post-release duplication or commentary.

We include results for the abalation experiments in Figure 5. Across benchmarks, the *Default* setting substantially outperforms *No Search* on HLE and SimpleQA, indicating that retrieval is a major driver of accuracy on these two tasks. The gap is especially pronounced for *Sonar Deep Research*, whose scores roughly triple on HLE and more than double on SimpleQA when retrieval is enabled. By contrast, GPQA is remarkably stable across all conditions, with only small, error-bar–sized fluctuations; this suggests that most GPQA questions can be correctly solved with the LLM's own capability without accessing online contents.

Source-specific Ablations (HF vs. the rest of the web). Blocking HuggingFace (*Blocked HF*) reduces performance relative to *Default* on HLE and SimpleQA, but the drops are modest and far smaller than the *Default–No Search* gaps. As is shown in Table 1, HuggingFace-based STC represents only a small portion across benchmarks, hence a relatively small overall drop by blocking HuggingFace domains is expected. This observation implies that while HF contributes to the gains, a large share of useful evidence (and thus potential STC) resides on *non*-HF domains (e.g., papers, blogs, mirrors).

[6] `https://docs.perplexity.ai/guides/date-range-filter-guide`. Per the Perplexity team, this filter relies on metadata available for the pages they index. Pages with missing metadata may not be filtered, so results can be incomplete for content published both before and after the target date.

Time ablation (Date Cutoff). Perhaps the most interesting observation is from applying the date filter. The *Date Cutoff* results remain well above *No Search* and *Only HF*, confirming that substantial pre-release information exists on the web. However, when compared to *Default*, scores are consistently lower on HLE and SimpleQA with particularly visible reductions for *Sonar Deep Research* on HLE. If the date filter implemented by Perplexity manages to create a complete indexing of the subset of webpages published prior to benchmark release, this indicates that post-release webpages (e.g. duplications, commentary, dataset examples, etc.) contribute meaningfully to accuracy (and perhaps STC) under the *Default* search setting. Nevertheless, on GPQA, date filtering has negligible impact, reinforcing that this dataset needs minimal to no online contents to solve its questions.

Overall Impact. In this section, we show that approximately 3% of evaluation data are affected by STC when used to assess search-based agents, and we quantify the resulting accuracy drop when contamination is mitigated through website blocking and others in additional ablation experiments. While 3% may seem small, it can be decisive for frontier benchmarks such as HLE, where even marginal differences influence state-of-the-art claims (e.g., o3 outperforms o4-mini by only 2.2%). Our conclusion is grounded more in *trustworthiness* than in raw performance: STC can occur with non-zero probability in the absence of proper safeguards, undermining the integrity of benchmarks. Moreover, STC erodes benchmark longevity. Once an agent's search logs are open-sourced or shared for research, benchmark data can leak beyond its original source. When millions of evaluation queries target the same benchmark, even small, repeated leaks can accelerate the benchmark's obsolescence, shortening its intended lifecycle.

4 Discussion

Capability Benchmarks Do Not Evaluate Search Capability. Current capability benchmarks are often created through a workflow involving human annotators. While humans may use web browsers or other online tools to locate or help derive the correct answers, the resulting questions are not meant to evaluate a model's capability of conducting web search. The reasons are three-fold. First, as the training set of modern LLMs can be as large as all the existing tokens on the Internet, the dataset curation process often involves creating new questions, or extrapolating and generalizing from existing ones to evaluate the model's capability to answer unseen questions through reasoning. Second, complex and difficult questions in more recent capability benchmarks do not necessarily require up-to-date knowledge that did not exist at benchmark creation time or hard-to-retrieve information hidden on the Internet. For example, SimpleQA acknowledges that all questions are solvable with knowledge existing before December 31, 2023 [17]. Third, there is no metric or rubric in current capability benchmarks that measure the capability of an LLM agent in searching, learning, and using retrieved information for problem-solving. Thus, hillclimbing only capability benchmarks might be a suboptimal way for improving the capability of search-based LLM agents. Constructing capability benchmarks with dynamic (e.g. time-varying answers) groudtruth labels is a follow-up research direction. We caution against the use of offline capability benchmarks as proxies for online model capability.

Towards Trustworthy Search-Time Evaluation. There is a set of existing interventions which can be applied in evaluations, but we stress their impact to stop STC is limited:

- **Canary strings:** Canary strings are used in datasets for model builders to detect and exclude from their training data, to avoid contamination. This is standard practice following work such as BIG-Bench [16]. However, they only work when everyone who distributes a dataset publicly also includes it, which as we observed in this study is not the case for many HuggingFace copies, even when the original dataset includes them (we even observed third-parties slicing out the dataset-embedded canary string in HLE and GPQA).
- **Dataset gating:** Benchmarks such as HLE and GPQA are already gated on HuggingFace, but as agents become more capable they may be able to bypass simple gating or encryption. Furthermore, this does not defend against ungated third-party distribution, which is the cause of the contamination we observe in this paper.
- **Single stage filtering:** Simply filtering out some set of URLs is not sufficient to address contamination fully. Our analysis hints at factors beyond HuggingFace that could also lead

to contamination. We recommend a multi-stage filtering model ("Swiss cheese model") to reduce contamination from multiple angles.

To enhance trustworthiness and mitigate STC in evaluating search-based agents, we propose the following recommendations:

- **Multiple search filters**. Implementing a comprehensive set of filters — similar to those used by the Perplexity API — can effectively control source sites during search operations, proactively preventing STC before it occurs.
- **Internal auditing**. We recommend establishing a multi-layered internal auditing system to detect STC incidents. Essential components include: (1) keyword filtering to identify major public websites that contain no relevant information for the problem-solving task (e.g., Huggingface model repositories); and (2) substring matching to detect exact questions from evaluation datasets within retrieved content. Additionally, LLM-based and human auditors can monitor agent trajectories to identify potential misuse of online information.
- **Transparency in reporting**. We advocate for responsible disclosure of evaluation setups, including detailed documentation of filter implementations and STC auditing processes. This transparency should encompass mitigation methodologies and their corresponding post-mitigation results. For open-source agents, releasing complete trajectories enables community-driven STC detection and reporting. For proprietary agents, providing trajectory abstracts can promote transparency and trustworthiness while preserving implementation security.

Limitations & Future Work We present this work as a preliminary finding of search-time contamination to bring attention to the issue. The main limitation of our work is that the simple, hard-coded URL detector does not capture the full extent of search-time contamination outside of the narrow range of HuggingFace sources named in a particular manner. Accordingly, an interesting avenue of future work is to use browser agents to audit every source retrieved to check for deeper contamination. It would also be interesting to explore if models know they are being evaluated (situational awareness [8]), cheating on the evaluation, or even encourage models to cheat (or encourage them to not cheat) on the evaluation, and whether this affects their ability to retrieve contaminated sources. Finally, we already requested takedowns/gating of a number of the third-party HLE HuggingFace uploads to preserve the benchmark integrity, which may impact the future reproducibility of this study.

5 Conclusion

In this work, we identify search-time contamination (STC) as a novel form of leakage in evaluating search-based LLM agents, where the retrieval step surfaces sources containing test questions alongside their answers, enabling agents to copy rather than genuinely reason. We demonstrate that approximately 3% of questions across HLE, SimpleQA, and GPQA are contaminated via HuggingFace sources, resulting in non-trivial accuracy gains that disappear when HuggingFace is blocked. Our findings reveal that traditional capability benchmarks are fundamentally unsuitable for evaluating search-augmented systems. Moving forwards, we propose best practices for evaluating search-based LLM agents, including prioritizing information-seeking benchmarks, implementing comprehensive source filtering, and establishing transparent contamination auditing to ensure trustworthy evaluation of search-based LLM agents.

Acknowledgment

We thank everyone on Scale AI's SEAL team for their feedback and ML infrastructure team for supporting the evaluations. We thank Huan Sun and Yu Su for their feedback to the early draft of the paper. We also thank Perplexity for debugging API errors we encountered over the course of our experiments.

Ethics

This paper's methodology and public dataset contain material that may enable malicious users to game the evaluations of search-based agents. While we recognize the associated risks, we believe it is essential to disclose this research in its entirety to help advance the integrity of agent benchmarks. Prior to release, we also disclosed our findings and the early draft of the paper to Perplexity.

References

[1] P. Chandrahasan, J. Jin, Z. Zhang, T. Wang, A. Tang, L. Mo, M. Ziyadi, L. F. R. Ribeiro, Z. Qiu, M. Dreyer, A. Asai, and C. Xiong. Deep research comparator: A platform for fine-grained human annotations of deep research agents. 2025. URL https://api.semanticscholar.org/CorpusID:280048405.

[2] G. DeepMind. Gemini deep research - your personal research assistant. URL https://gemini.google/overview/deep-research/.

[3] C. Deng, Y. Zhao, Y. Heng, Y. Li, J. Cao, X. Tang, and A. Cohan. Unveiling the spectrum of data contamination in language models: A survey from detection to remediation, 2024. URL https://arxiv.org/abs/2406.14644.

[4] X. Deng, Y. Gu, B. Zheng, S. Chen, S. Stevens, B. Wang, H. Sun, and Y. Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114, 2023.

[5] Y. Dong, X. Jiang, H. Liu, Z. Jin, B. Gu, M. Yang, and G. Li. Generalization or memorization: Data contamination and trustworthy evaluation for large language models, 2024. URL https://arxiv.org/abs/2402.15938.

[6] M. Du, B. Xu, C. Zhu, X. Wang, and Z. Mao. Deepresearch bench: A comprehensive benchmark for deep research agents. *ArXiv*, abs/2506.11763, 2025. URL https://api.semanticscholar.org/CorpusID:279391682.

[7] B. Gou, Z. Huang, Y. Ning, Y. Gu, M. Lin, W. Qi, A. Kopanev, B. Yu, B. J. Gutiérrez, Y. Shu, C. H. Song, J. Wu, S. Chen, H. N. Moussa, T. Zhang, J. Xie, Y. Li, T. Xue, Z. Liao, K. Zhang, B. Zheng, Z. Cai, V. Rozgic, M. Ziyadi, H. Sun, and Y. Su. Mind2web 2: Evaluating agentic search with agent-as-a-judge, 2025.

[8] R. Laine, B. Chughtai, J. Betley, K. Hariharan, M. Balesni, J. Scheurer, M. Hobbhahn, A. Meinke, and O. Evans. Me, myself, and ai: The situational awareness dataset (sad) for llms. *Advances in Neural Information Processing Systems*, 37:64010–64118, 2024.

[9] M. Miroyan, T.-H. Wu, L. King, T. Li, J. Pan, X. Hu, W.-L. Chiang, A. N. Angelopoulos, T. Darrell, N. Norouzi, and J. Gonzalez. Search arena: Analyzing search-augmented llms. *ArXiv*, abs/2506.05334, 2025. URL https://api.semanticscholar.org/CorpusID:279243096.

[10] OpenAI, . URL https://openai.com/index/introducing-deep-research/.

[11] OpenAI, . URL https://cdn.openai.com/pdf/8124a3ce-ab78-4f06-96eb-49ea29ffb52f/gpt5-system-card-aug7.pdf.

[12] Perplexity. URL https://www.perplexity.ai/hub/blog/introducing-perplexity-deep-research.

[13] L. Phan, A. Gatti, Z. Han, N. Li, J. Hu, H. Zhang, C. B. C. Zhang, M. Shaaban, J. Ling, S. Shi, et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

[14] D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, and S. R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

[15] Y. Song, K. Thai, C. M. Pham, Y. Chang, M. Nadaf, and M. Iyyer. Bearcubs: A benchmark for computer-using web agents. *ArXiv*, abs/2503.07919, 2025. URL https://api.semanticscholar.org/CorpusID:276929324.

[16] A. Srivastava, A. Rastogi, A. Rao, A. A. Shoeb, A. Abid, A. Fisch, A. R. Brown, A. Santoro, A. Gupta, A. Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on machine learning research*, 2023.

[17] J. Wei, N. Karina, H. W. Chung, Y. J. Jiao, S. Papay, A. Glaese, J. Schulman, and W. Fedus. Measuring short-form factuality in large language models. *arXiv preprint arXiv:2411.04368*, 2024.

[18] J. Wei, Z. Sun, S. Papay, S. McKinney, J. Han, I. Fulford, H. W. Chung, A. T. Passos, W. Fedus, and A. Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025.

[19] XAI. URL https://x.ai/news/grok-4.

[20] Y. Xi, J. Lin, M. Zhu, Y. Xiao, Z. Ou, J. Liu, T. Wan, B. Chen, W. Liu, Y. Wang, R. Tang, W. Zhang, and Y. Yu. Infodeepseek: Benchmarking agentic information seeking for retrieval-augmented generation. *ArXiv*, abs/2505.15872, 2025. URL https://api.semanticscholar.org/CorpusID:278788862.

[21] H. Zhang, J. Da, D. Lee, V. Robinson, C. Wu, W. Song, T. Zhao, P. Raja, C. Zhuang, D. Slack, et al. A careful examination of large language model performance on grade school arithmetic. *Advances in Neural Information Processing Systems*, 37:46819–46836, 2024.

A Substring Checker

Python Code: HuggingFace Contamination Checker

```
def check_hf_contam(sources: list[str], substrings: list[str]) -> str:
    """
    Checks whether any of the URLs corresponding to a
    single sample is HuggingFace contaminated.

    Args:
        sources: Sources corresponding to a single sample.
        substrings: Substrs which count as contamination when in the URL.

    Returns:
        The first source of contamination, "" if no HF contam detected.
    """
    for source in sources:
        if 'huggingface' in source:
            if any(substring in source for substring in substrings):
                return source
    return ""

# Contamination substring definitions
HLE_SUBSTRINGS = ['hle']
SIMPLEQA_SUBSTRINGS = ['simpleqa', 'simple_qa', 'simple-qa']
GPQA_SUBSTRINGS = ['gpqa']
```