SEARCHING IN THE DARK: HOW DO SEARCH-AUGMENTED LLMS HANDLE ABSTRACT QUERIES?

Anonymous authors
Paper under double-blind review

ABSTRACT

Current search benchmarks largely focus on well-specified, multi-constraint queries that determine a target answer. However, in practice, users often omit constraints and may describe what they are looking for indirectly. We study abstract search: queries that leave multiple plausible answers, and whose available constraints may be expressed explicitly or described indirectly. We construct a dataset of 515 queries spanning two search settings: 255 human-authored *Entity Recall* queries, where a user has a target entity in mind but provides partial or indirect cues that could fit other candidates and 260 *Open Discovery* queries, where users state their criteria directly but those criteria include multiple valid candidates. Across seven search-augmented LLMs, we find that models struggle to balance retrieval breadth and candidate validity: most models either retrieve too conservatively and miss valid candidates or retrieve too broadly and admit invalid candidates, with no model achieving both high precision and coverage. We observe that models struggle to resolve indirect descriptions in underspecified queries, with the strongest model recovering only 67.5% of user-intended entities in Entity Recall and indirect descriptions are associated with the largest drop in retrieval performance. Models also remain overconfident: Web search improves calibration relative to parametric memory, but this improvement comes from accuracy rising towards confidence, not from models becoming more cautious. Overall, our findings highlight the need to characterise search behaviour beyond well-specified settings and reveal important blind spots that emerge when users search abstractly.

1 INTRODUCTION

Search-augmented language models and deep research systems are now widely used for everyday information seeking (Zhang et al., 2025; Miroyan et al., 2026; Khan et al., 2026). Recent benchmarks evaluate search capabilities using well-specified, often multi-constraint queries whose answer or answer set is predetermined during dataset construction (Wei et al., 2025; Cheng et al., 2025; Wong et al., 2026; Gupta et al., 2026; CH-Wang et al., 2025). In practice, however, users do not always provide enough information to determine what should be returned. Users search abstractly: underspecifying their needs (Spärck-Jones et al., 2007; Malaviya et al., 2025), using partial descriptions rather than precise identifiers (Arguello et al., 2021), or expressing what they seek through indirect concepts (Furnas et al., 1987; Keyvan & Huang, 2022). Recent work has recognised this and evaluates whether models can reduce uncertainty by asking for clarification (Deng et al., 2026; Tao et al., 2026; Huang et al., 2026). But clarification assumes that additional user information can narrow the query enough to identify what should be returned. This is not always possible: a user may have no further information to provide, as when half-remembering a book or film (Arguello et al., 2021), or the information need itself may admit multiple valid answers, as in exploratory search (Malaviya et al., 2023). This leaves an important setting largely untested: search when uncertainty remains in the query itself. In this setting, models must decide how to act despite that uncertainty: whether they can resolve indirect clues, balance retrieval breadth against candidate validity, and calibrate confidence when several answers remain plausible.

To close this gap, we introduce *abstract search*: search where uncertainty remains in the query at retrieval time. We characterise queries along two dimensions: the *degree of specification*, which

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

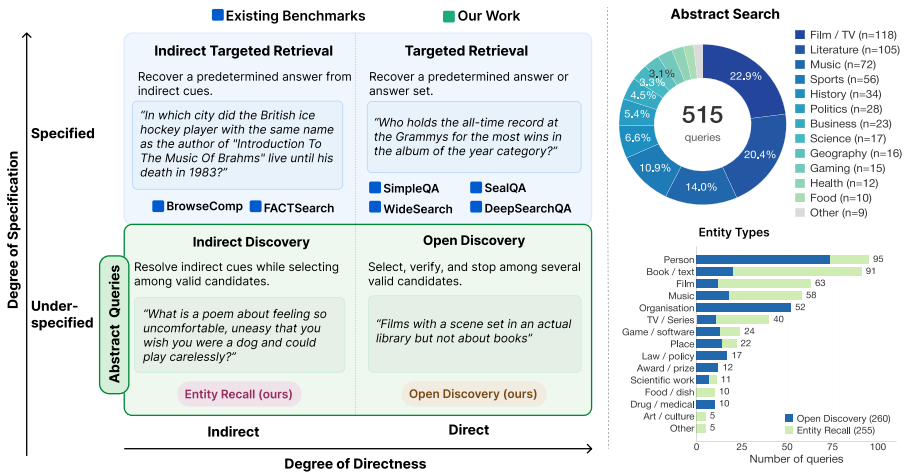


Figure 1: **Abstract search setting and dataset overview.** We study abstract search: underspecified queries that may express constraints directly or indirectly and leave multiple candidates valid. Our dataset contains 515 such queries spanning diverse domains and entity types.

captures whether the stated constraints determine what should be returned, whether a single answer or a set of answers, and the *degree of directness*, which captures whether those constraints are stated explicitly or described indirectly and must be interpreted during retrieval. Figure 1 shows the resulting taxonomy. We study two settings: Entity Recall, where a user has a target entity in mind but describes it through partial or indirect clues that may also match other candidates; and Open Discovery, where the stated criteria admit multiple valid candidates. Together, these settings let us study both cases where an intended target remains difficult to recover and where no single target exists at all. We construct 515 diverse queries: 255 Entity Recall queries through human red-teaming from known seed entities, and 260 Open Discovery queries through a synthetic generation pipeline grounded in observed real-world behaviour. Because abstract queries leave the candidate space open, constructing an exhaustive gold set would itself require solving the search problem. We therefore evaluate returned candidates against query-specific criteria using an automated verifier; for Entity Recall, we additionally measure whether the model recovers the user's intended entity. This lets us separately measure target recovery, candidate validity, retrieval breadth, and confidence when uncertainty remains in the query.

We evaluate seven search-augmented LLMs and find three consistent limitations. First, models struggle to recover the entity a user has in mind when clues are incomplete or indirect: even the strongest model recovers the intended entity on only 67.5% of Entity Recall queries. Second, models struggle to balance retrieval breadth and candidate validity: most either retrieve too conservatively, sacrificing Coverage, or retrieve broadly and include invalid candidates, sacrificing Precision. Third, models remain overconfident under underspecification: Web search reduces uncertainty about the underlying facts, but not the uncertainty introduced by the query itself. We publicly release the dataset, code, and evaluation traces.

2 Operationalising Abstract Queries

We study the use case where a user expresses an information need as a natural-language query to a Web-search-augmented language model. As mentioned, we characterise a search query along two dimensions. First, the **degree of specification**: whether the stated constraints determine an answer or answer set. Second, the **degree of directness**: how explicitly those constraints identify the entities, concepts, or attributes needed for retrieval. We define each property below before combining them into a taxonomy:

Specification describes how far the constraints provided in a query narrow the answer. A query is *specified* when its stated constraints determine a single target answer, which could be one item or a list of items (e.g., 'Which actors have played James Bond?'). These constraints may include named

entities, geographical anchors, time periods, or multiple pieces of information that together narrow the space of possible answers (Jiang et al., 2026; Malaviya et al., 2025). A query is *underspecified* when its stated constraints admit several different valid answers. Specification is not determined by the number of answers a query requests. A query may ask for a single item or a set of items and but still be specified if its constraints determine the target. At the same time, specification is a property of the query as written and presented to the system, not of the user's latent intent. A user may have a target in mind while descrubing the constraints in a way that several different answers are valid.

Directness describes how explicitly the stated constraints identify the entities, concepts, or attributes needed for retrieval. A query is *direct* when its constraints identify these elements explicitly. A query is *indirect* when one or more constraints are expressed through descriptions (Ravfogel et al., 2024), latent concepts (Tchuindjo et al., 2026), or relational chains (Yang et al., 2018) that the system must first infer or identify during retrieval. Our notion of indirectness builds on longstanding work on vocabulary mismatch (Furnas et al., 1987), indirect referring expressions (Hosseini et al., 2023), known-item search (Buckland, 1979; Lee et al., 2006), and tip-of-the-tongue retrieval (Arguello et al., 2021).

We treat **specification** and **directness** as independent properties. A query may be indirect yet still determine a single answer. A query may state its constraints directly yet remain underspecified, leaving several answers valid. Individual constraints in a query may also differ: some may be expressed directly while others require resolving, and some may substantially narrow the answer while others barely narrow it. We therefore assess both properties over the query as a whole. A query is *underspecified* when its constraints taken together leave several answers valid, and *indirect* when satisfying the query requires resolving at least one constraint. Combining these two properties gives the four-cell taxonomy in Figure 1. We describe each setting below, along with the benchmarks that occupy it:

- **Targeted Retrieval (specified, direct).** The constraints determine what should be returned, and the query states them directly. These range from single-answer factual lookups, as in SimpleQA (Wei et al., 2024a) and SealQA (Pham et al., 2026), to queries that require a complete answer set, as in WideSearch (Wong et al., 2026) and DeepSearchQA (Gupta et al., 2026). These queries test whether a system can satisfy a well-defined information request: recovering the answer or answer set determined by the query, without having to decide among several equally valid candidates or resolve indirect expressions.

- **Indirect Targeted Retrieval (specified, indirect).** The constraints determine what should be returned, but one or more constraints are expressed indirectly. Benchmarks such as BrowseComp (Wei et al., 2025) and FACTSearch (Cheng et al., 2025) make targeted retrieval more challenging by obscuring information that would otherwise identify the answer directly. They do this by replacing directly identifying information with vague descriptions, approximate references, or relational chains that must be resolved during retrieval. These queries test whether a system can resolve indirect expressions through search and persist until it recovers the determined answer.

- **Open Discovery (underspecified, direct).** The constraints leave several different answers valid, and the query states the constraints directly. These queries are common in exploratory search, where a user provides a set of criteria without having specific candidates in mind and expects the system to find candidates which satisfy them. Unlike targeted retrieval, the query does not determine which valid candidates the system should return. The system must therefore decide which candidates to explore, verify whether they satisfy the stated constraints, determine how broadly to search, and decide when to stop. These queries test how a system balances the breadth of its search and validity of the candidates when the query leaves the candidate choice open.

- **Indirect Discovery (underspecified, indirect).** The constraints leave several candidates valid, and the query express one or more of the constraints indirectly. For example, a user may have a particular book in mind but remember only parts of its plot without remembering specific character names or titles. The resulting description may require the system to resolve indirect cues while still matching several different books that could fit the description. These queries test whether a system can resolve indirect expressions, as in Indirect Targeted Retrieval, while balancing search breadth and validity, as in Open Discovery. BLUR (CH-Wang et al., 2025) is closely related in its use of tip-of-the-tongue descriptions, but assumes a single target answer.

In this work, we study *abstract search queries*, an umbrella term for the underspecified setting in our taxonomy. This lets us ask how systems behave when the query itself leaves uncertainty unresolved, including how they calibrate search breadth, maintain candidate validity, and respond to indirect expressions.

How do we evaluate abstract search? Abstract queries can admit multiple valid candidates, making an exhaustive gold answer set difficult to construct: enumerating one requires solving the open-ended search problem itself, while enforcing exhaustiveness may introduce constraints not stated in the query. We therefore evaluate each returned candidate against a set of query-specific criteria. For each query q, we define a set of criteria $\mathcal{C}(q) = \{c_1, \ldots, c_m\}$. Each criterion is *faithful* (Wanner et al., 2024), corresponding to a constraint expressed in the query without introducing an additional requirement; *atomic* (Min et al., 2023), expressing a single verifiable condition; and *verifiable* (Song et al., 2024), such that whether a candidate satisfies it can be checked using Web evidence. Collectively, the criteria cover the stated constraints relevant to candidate validity. A returned candidate x is valid iff it satisfies every criterion, i.e., $V(x, q) = 1 \iff \forall c_i \in \mathcal{C}(q),\ c_i(x) = 1$. At evaluation time, a Web-enabled verifier is given the query, a returned candidate, and its criteria, and searches for evidence to determine whether the candidate satisfies each criterion. This exploits an asymmetry of verification in retrieval (Wei, 2026; Zeng et al., 2026): checking whether a given candidate satisfies the stated constraints is easier than enumerating every candidate.

3 Dataset Construction

We construct two datasets for the abstract search setting, each designed around four properties. Queries should be *realistic*, reflecting plausible user information needs; *diverse*, spanning different domains and entity types; *non-trivial*, requiring deep and wide search across multiple sources; and *verifiable*, with objective criteria that can be checked against information available on the Web. For Indirect Discovery, we collect Entity Recall queries through human red-teaming, where authors iteratively refine human-written queries against a Solver model. For Open Discovery, we use a synthetic-generation pipeline grounded in real-world search behaviour to generate open-ended, entity-seeking queries. Our final *abstract search* dataset contains 515 queries: 255 Entity Recall queries and 260 Open Discovery queries. We describe each dataset construction approach below.

Entity Recall We construct Entity Recall queries through a **human red-teaming approach**. A human annotator starts from a known seed entity and iteratively rewrites a query against a Web-enabled solver, removing or rewriting clues that make the seed easy to recover. This continues until the solver fails to retrieve the seed within a fixed calibration budget. After finalising the query, the annotator writes or verifies criteria generated corresponding to the clues expressed in it. We then apply automated filtering, removing queries where the seed entity and query description do not match and dropping or repairing any criteria that do not faithfully reflect the query. We further apply a parametric-recall screen to remove queries recoverable from parametric memory alone. For each query, we sample five responses from GPT-5.4 with Web search disabled, asking for a single answer rather than a list of candidates, and discard queries where the seed entity is returned in any sampFinally, the authors manually review each query, removing cases where the stated constraints appear to uniquely determine the seed and checking that the seed is valid and the criteria are faithful and Web-verifiable. The seed must satisfy the query but need not be the only valid answer. We recruit annotators through Prolific and to ensure geographic coverage, we collect data across multiple annotation rounds with different geographic filters. Full annotation instructions and model details are provided in Appendix B. Annotators spend a median of 20.9 minutes per query (mean 29.3) and an of average ~ 4 query revisions against the solver, reflecting the iterative effort required to construct each query.

We collect 352 candidate queries; automated filtering removes 50, and final author review removes a further 47, leaving 255 queries. Query length varies widely, with a median of 62 words (mean 71), with some queries exceeding 200 words. Queries contain an average of 6.34 criteria, ranging from 2 to 12. The final set spans 22 entity types, 41 countries and 39 languages, reflecting the breadth of entities represented in the dataset. To vary the difficulty of red-teaming, queries are tested against four Solvers: GPT-5.4-mini (82 queries), GPT-5.6 Luna (74), GPT-5.4 (64), and GPT-5.6 Sol (35). Queries calibrated against stronger solvers are harder for the evaluated models (Table 10). We use an LLM to annotate each revision by comparing consecutive query versions and assigning one or

more edit types at each turn; full annotation details are provided in Appendix B. Annotators *under-specify* queries by removing specific details (54.3%), dropping named entities (18.7%), or widening temporal references (11.2%); *specify* them by adding details (40.4%); *abstract* them by describing details indirectly (32.1%) or hedging remembered information (7.5%); and *add distractors* (8.8%).

Open Discovery We use **a synthetic-generation pipeline** grounded in real search behaviour to construct Open Discovery queries. Our pipeline begins with an input seed grounded in Open Discovery queries collected from SearchArena (Miroyan et al., 2026) which guides a multi-turn Generator–Solver loop. The resulting queries then pass through automated filtering and human verification. Each seed specifies a domain, a list of entity types and a target search capability that the generated query is intended to test. We target four search capabilities: *cross-domain*, which requires combining facts about the same candidate from sources that are not indexed together; *negative-evidence*, which requires confirming the absence of a property within a defined scope; *comparative*, which requires comparing a candidate within a defined peer group; and *nested sequence*: where verifying one criterion requires resolving an ordered sequence from another criterion. We provide examples for each search capability in Tanl C. Given a seed, a web-enabled Generator browses the web and writes an initial query together with criteria needed to verify candidates. A second web-enabled agent, the Solver, searches for shortcuts, such as a single page that could easily resolve the query. If the Solver finds such a shortcut, it returns the supporting evidence and feedback to the Generator, which revises the query before it is tested again. We set a minimum and maximum number of Generator–Solver turns, after which the loop stops if the Solver cannot find an easy way to answer the query. Candidate queries then pass through automated filters applied by a separate model. We drop queries that collapse to a specified answer, that are subjective or cannot be verified using information available on the web or that do not capture the intended target search capability. We manually check each criterion against its query, repairing those that are unfaithful and dropping those that cannot be repaired. Finally, we review each query and its criteria, keeping only queries that are genuinely open-ended, whose criteria are faithful to the query and can be verified with information available on the Web and ensure the intended target search capability is present. We used GPT-5.4 and GPT-5.4-mini in the Generator–Solver loop, and GPT-5.4 for automated filtering. Full prompts and implementation details are provided in Appendix C. We generated 906 queries, of which automated filtering removed 435 and author review removed 211 further. The resulting open discovery dataset contains 260 queries, comprising 71 cross-domain, 68 nested sequence, 62 negative-evidence, and 59 comparative queries. The queries are relatively short compared to entity recall, with a median length of 27 words per query. They contain between 3 and 10 criteria, with a mean of 6 criteria per query. We provide full details of the construction approach in Appendix C.

4 Experimental Setup

Models. We evaluate seven LLMs spanning five model families: GPT-5.6 Sol (OpenAI, 2026), GPT-5.6 Luna (OpenAI, 2026), Gemini 3.7 Flash (Google DeepMind, 2026), Claude Opus 5 (Anthropic, 2026a), Claude Sonnet 5 (Anthropic, 2026b), DeepSeek V4 Pro (DeepSeek AI, 2026), and Qwen 3.8 Max (Qwen Team, 2026). We run all models with reasoning effort set to *high*, unless specified otherwise. We focus on model-level search behaviour rather than end-to-end deep-research or multi-agent systems, allowing observed differences to be attributed more directly to the underlying model. We provide the exact model versions, inference settings in Appendix D.1.

Search Harness. We evaluate all models using the same search harness, with Exa[1] for Web search and Jina[2] for reading web pages. Each Exa search returns up to 10 results together with a short highlighted excerpt from each page, after which the model may choose which pages to read. We use a shared harness to control for differences in model-native search providers and allow us to compare model-specific search behaviour under the same retrieval environment.

Evaluation Protocol. We evaluate both tasks in the same harness with task-specific prompts. We prompt models to return a list of candidate entities, each with a verbalised confidence score from 0–100 and a short evidence statement. We do not specify how many candidates to return or where to search, since search breadth and source selection are part of the behaviour we evaluate. Each trajectory is capped at 100 LLM turns; if the limit is reached, the model is prompted to answer from

[1] https://exa.ai/
[2] https://jina.ai/

Model	Hit Rate (%)		Precision (%) ↑	Coverage@4 (%) ↑	F1@4 (%) ↑	Candidates (avg.)	Searches (avg.)
	w/o Search	w/ Search ↑					
GPT-5.6 Sol	53.33	**67.45**	**58.37**	34.71	**37.38**	4.17	39.13
Claude Opus 5	40.39	64.71	31.08	**39.31**	32.94	6.54	12.10
Gemini 3.7 Flash	48.24	61.18	46.53	32.55	34.12	3.85	11.15
DeepSeek V4 Pro	27.06	53.33	37.91	32.65	32.24	4.66	54.58
GPT-5.6 Luna	36.86	47.84	52.73	27.75	32.29	2.70	59.25
Claude Sonnet 5	20.78	38.04	24.82	26.37	23.87	5.19	9.44
Qwen 3.8 Max	26.67	36.86	30.06	26.18	25.49	4.29	21.54

Table 1: Entity Recall **results, ordered by Hit Rate with search, our primary metric for this setting**. Hit Rate without search is measured with search tools disabled. Coverage@4 and F1@4 use $K = 4$, the median number of candidates returned across systems and queries. Bold denotes the highest value in each search-enabled performance column. Candidates and Searches are per-query means.

the trajectory so far, although this occurs rarely (Table 9). We place no separate limit on search or page-read calls. All evaluations are run within the same time window to minimise variation in Web content and search results. For the parametric baseline, we use the same prompts and inference settings with search tools disabled. Full prompts are provided in Appendix D.3.

Criteria-based Evaluation. We apply the criteria-based evaluation described in Section 2 to every candidate returned in both settings. Given a query, its criteria, and a returned candidate, a verifier searches the Web and labels whether the candidate satisfies each criterion as YES or NO; partial matches are labelled NO. We use Gemini 3.7 Flash as a cost-efficient verifier for large-scale criterion checking [3] , using the same search harness as in evaluation. To validate these judgements, three independent annotators annotate 166 candidate–criterion pairs across 30 queries. Agreement between the human annotations and the verifier's labels is 0.904 raw exact agreement and 0.780 unweighted Cohen's κ, indicating substantial agreement, and matches the level the annotators reach among themselves (0.859). Full verifier prompts and validation details are provided in Appendix E. For Entity Recall, where each query has a known seed entity, we use GPT-5.6 Luna to determine whether the seed entity appears among the returned candidates.

Evaluation Metrics. We adopt standard information-retrieval metrics and evaluate systems using Precision, Coverage@K, F1@K, and Hit Rate. We compute each metric per query and macro-average across queries. **Precision** measures the fraction of returned candidates that are valid. **Coverage@K** measures whether enough valid candidates are returned. Because the complete set of valid candidates is unobservable, we follow Wei et al. (2024b) and treat K as a target number of valid candidates for a useful response. We set K to the median number of candidates returned across all systems and queries within each setting, giving $K = 5$ for Open Discovery and $K = 4$ for Entity Recall. We additionally report results across a range of K values in Appendix H. **F1@K** is the harmonic mean of Precision and Coverage@K, rewarding systems that return both a high proportion and a sufficient number of valid candidates. **Hit Rate** applies only to Entity Recall and measures whether the known seed entity appears among the returned candidates. We rank systems by F1@K for Open Discovery, where both candidate quality and breadth matter, and by Hit Rate for Entity Recall, where success requires recovering the intended entity.

5 RESULTS & DISCUSSION

Table 1 reports Entity Recall results and Table 2 reports Open Discovery results. We discuss the findings below.

Finding 1: Models struggle to resolve indirect descriptions in underspecified queries. GPT-5.6 Sol achieves the highest Hit Rate in Entity Recall (Table 1) at 67.45%, followed by Claude Opus 5 at 64.71% and Gemini 3.7 Flash at 61.18%. Substantial headroom remains, as even the

[3] Across both tasks the verifier issues 123,815 criterion judgments over 20,817 candidates, each requiring its own web evidence and cited source.

Model	Precision (%) ↑	Coverage@5 (%) ↑	F1@5 (%) ↑	Candidates (avg.)	Searches (avg.)
GPT-5.6 Sol	**65.05**	56.62	**56.50**	8.01	82.93
Gemini 3.7 Flash	61.12	50.92	52.02	5.37	21.89
Claude Opus 5	48.59	**56.77**	50.67	7.92	13.23
DeepSeek V4 Pro	59.98	45.00	47.99	5.33	62.72
GPT-5.6 Luna	56.21	45.92	46.69	6.53	95.67
Qwen 3.8 Max	39.82	54.62	44.62	10.25	27.01
Claude Sonnet 5	37.00	38.15	36.53	5.86	13.98

Table 2: Open Discovery **results, ordered by F1@5.** Coverage@5 and F1@5 use $K = 5$, the median number of candidates returned across systems and queries. Bold denotes the highest value in each performance column. Candidates and Searches are descriptive averages per query.

strongest model misses roughly one third of target entities. Comparing each annotator's initial query with its final version, Hit Rate falls from 77% to 56% (Figure 2). Describing a previously explicit detail indirectly is associated with one of the largest drops, comparable to removing the detail even though the information is preserved, whereas adding a detail produces one of the smallest drops. Although edit types co-occur (Appendix B), this pattern suggests that Entity Recall is difficult not only when users provide less information, but also when they express it indirectly. Web search improves Hit Rate for every model, by 16.6 percentage points on average compared to without search. Hit Rates with and without search are strongly correlated (Spearman $\rho = 0.89$) suggesting that search complements rather than replaces parametric knowledge: models with stronger parametric knowledge may be better able to resolve indirect expressions. A human oracle is difficult to define for Entity Recall, since success depends partly on prior linguistic or cultural familiarity with the intended entity. We therefore contextualise the best single-model Hit Rate of 67.5% with a model oracle: counting a query as solved if any of the seven models recovers the seed entity gives 84.3%, 16.8 points higher, and leaves 15.7% of queries that no model solves. We provide five annotated queries on which no model recovers the user's intended entity, though several return close matches (Appendix I).

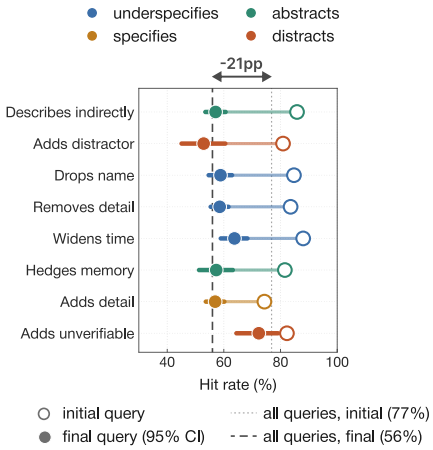


Figure 2: **Indirect descriptions are associated with among the largest drops in Hit Rate.** Hit Rate on each annotator's initial query (hollow) and final version (filled, 95% CI), grouped by edit type and coloured by edit category.

Finding 2: Models struggle to balance retrieval breadth and candidate validity. In Open Discovery (Table 2), GPT-5.6 Sol achieves the highest F1@5 (56.6%), combining the highest Precision (65.1%) with near-best Coverage@5 (56.7%). Other models separate more clearly: Gemini 3.7 Flash and DeepSeek V4 Pro return fewer candidates with higher Precision but lower Coverage, while Claude Opus 5 and Qwen 3.8 Max return more candidates with higher Coverage but lower Precision. A similar trade-off appears in Entity Recall (Table 1): GPT-5.6 Sol has the highest Precision (58.4%), while Claude Opus 5 has the highest Coverage@4 (39.3%) and remains highest across K (Appendix H). Overall, most models either retrieve too conservatively, sacrificing Coverage, or retrieve broadly and include invalid candidates, sacrificing Precision.

Finding 3: Cross-domain queries are the hardest across every model. Cross-domain queries require combining evidence about the same candidate from sources that cover different domains. They yield the lowest F1@5 for every model (Figure 3), averaging 41.2%, compared with 54.7% for comparative queries, the highest-performing capability. Every model also searches more than its own median on cross-domain queries, yet these queries return the fewest candidates on average (5.65, versus 6.86–8.12 for the other capabilities). Models therefore search more but surface fewer

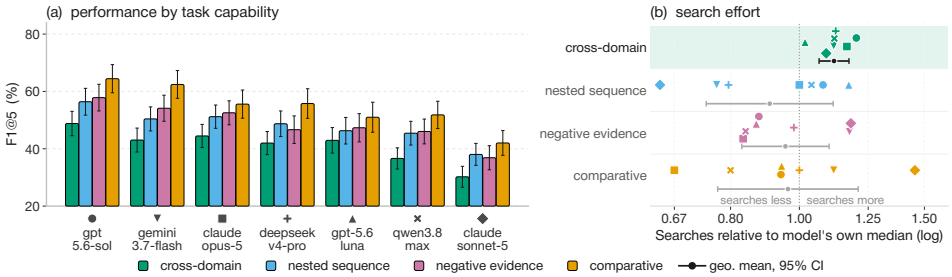


Figure 3: **Cross-domain queries are the hardest despite the most search effort.** (a) Cross-domain(■) yields the lowest F1@5 for every model. (b) Every model searches above its own median on cross-domain queries; on other capabilities, models fall on both sides.

candidates, suggesting that the bottleneck is not search effort itself, but finding candidates that satisfy constraints across domains. A full breakdown is provided in Appendix F.

Finding 4: Search reduces knowledge uncertainty, but models remain overconfident under query uncertainty. Entity Recall offers a natural test of calibration because uncertainty has two sources: the user's description may be incomplete or indirect, leaving several plausible entities even with perfect knowledge, or the model may lack the knowledge to recover the target, which Web evidence can reduce. We evaluate calibration both when models rely only on their parametric knowledge and when they can retrieve evidence from the Web. For each query, we check whether the highest-confidence candidate matches the seed entity. We refer to the fraction of queries where it does as observed accuracy, and compare this with stated confidence using Expected Calibration Error (ECE). Figure 4 shows the corresponding reliability curves. Because the queries may admit valid candidates other than the seed, we additionally evaluate calibration against criterion-level candidate validity in Appendix G. In both settings, most confidence bins lie below the diagonal (Figure 4), indicating that stated confidence exceeds observed accuracy. Search increases confidence for every model, but it also increases observed accuracy; for six of seven models, the gain in accuracy is larger than the gain in confidence, narrowing the gap between the two and reducing ECE (Table 12). For example, ECE falls from 39.9% to 33.1% for GPT-5.6 Sol, from 45.9% to 35.6% for Gemini 3.7 Flash, and from 38.9% to 24.3% for DeepSeek V4 Pro. We note that the improvement in calibration comes from accuracy catching up with already-high confidence rather than from models becoming more cautious when given external evidence. Overall, Web search reduces uncertainty arising from gaps in parametric knowledge, but models remain overconfident about having recovered the user's intended entity, suggesting that they still fail to reflect uncertainty left by an underspecified query.

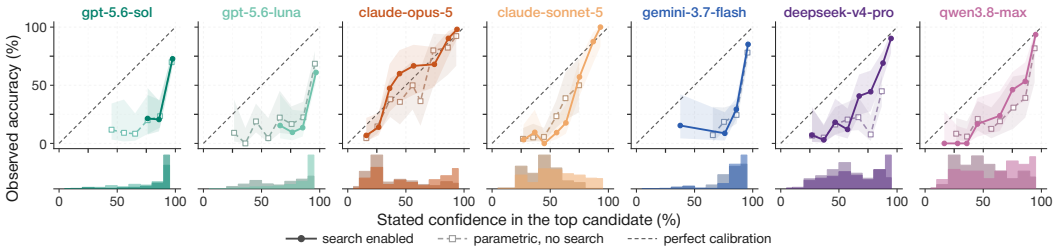


Figure 4: **Models are overconfident with and without search.** Most curves lie below the diagonal and confidence clusters near 100% (bottom strips). Search (solid) raises accuracy over parametric (dashed) but rarely closes the gap. Bands are 95% Wilson intervals over 10 equal-width bins.

Finding 5: Wikipedia dominates model reads, while search models exhibit source priors. Wikipedia appears in 42.6% of Open Discovery reads and 32.3% of Entity Recall reads. In Open Discovery, `site:` requests are often topic-specific: 56.2% of business requests target `sec.gov`, while 28.7% of science requests target `ptacts.uspto.gov`. In Entity Recall, Reddit accounts for 21.7% of all `site:` requests across all seven models, most often through communities such as `r/tipofmytongue`, `r/whatsthatbook`, and `r/tipofmyjoystick`, which are dedicated to identifying half-remembered entities. Per-model and per-task source breakdowns are shown in Appendix J.

We provide a diversity analysis in Appendix K, measuring how much models overlap in the candidates they return and how varied each model's own candidates are.

6 Related Work

Underspecified and indirectly expressed information-seeking queries are well-studied problems in information retrieval. Classic work on ambiguous requests examines how to evaluate search systems when queries admit multiple interpretations (Spärck-Jones et al., 2007; Clarke et al., 2009), while work on vocabulary mismatch, known-item retrieval, and tip-of-the-tongue retrieval examines how users search using partial or indirect descriptions (Furnas et al., 1987; Arguello et al., 2021; Bhargav et al., 2022). More recent evaluation approaches bring these problems to LLMs and search models. AmbigQA (Min et al., 2020) studies ambiguous questions by requiring systems to recover multiple plausible answers and the interpretations associated with them, while InteractComp (Deng et al., 2026), DiscoBench (Tao et al., 2026), and Clarify-Then-Search (Huang et al., 2026) evaluate whether search models can identify missing or ambiguous information and seek clarification. These approaches use clarification to reduce ambiguity and help identify the user's target. However, clarification is not always possible or necessary: users may genuinely have no additional clarification to provide in known-item search (Arguello et al., 2021), and some information needs require multiple valid answers rather than a single target (Malaviya et al., 2023). We therefore hold the query fixed and evaluate how search models operate under the residual uncertainty that remains when users provide no additional information and multiple valid candidates remain on the Web. Other benchmarks retain a predetermined target while making it difficult to recover: BrowseComp (Wei et al., 2025) obscures identifying information, FACTS Search (Cheng et al., 2025) requires multi-hop Web retrieval, and BLUR (CH-Wang et al., 2025) evaluates tip-of-the-tongue recovery from partial recollections. Entity Recall shares this known-target construction: annotators begin with a seed entity in mind and write a query describing it. However, we do not require the query to uniquely identify the seed; other candidates may also satisfy the constraints. This lets us to separately evaluate whether a model recovers the user-intended entity and whether the candidates it returns are valid. WideSearch (Wong et al., 2026) and DeepSearchQA (Gupta et al., 2026) evaluate exhaustive retrieval against a predetermined answer set. We instead leave the valid candidate set open, requiring models to balance retrieval breadth against candidate validity rather than exhaustively recover a predefined set. QUEST (Malaviya et al., 2023) is closest to Open Discovery, but uses a fixed Wikipedia corpus with enumerable valid entities; we instead study the open Web, where the candidate set remains open.

7 Conclusion

We study abstract search: real-world information-seeking needs where users underspecify what they are looking for, such that multiple candidates may plausibly satisfy the query, while expressing the constraints either directly or indirectly. We introduce 515 abstract-search queries and a criteria-based evaluation framework for settings where a gold-reference set is infeasible to define. Across seven search-augmented LLMs, we find that models handle this uncertainty poorly: they struggle to balance how broadly to search against the validity of retrieved candidates, have difficulty recovering user-intended entities from partial or indirect cues, and remain overconfident under underspecification. This matters for how current search models are evaluated. Many current search benchmarks assume that the query determines a target answer and therefore measure whether a model recovers it. Our results show that this can obscure important differences in how search models behave. On Entity Recall, for example, GPT-5.6 Sol and Claude Opus 5 achieve similar Hit Rates, yet exhibit different behaviours: Sol retrieves fewer candidates with substantially higher precision, while Opus retrieves more broadly and is substantially better calibrated. Our findings also raise a broader question for training: whether RL objectives that reward recovering a gold answer (Song et al., 2025; Chu et al., 2026; Li et al., 2026; Bashir et al., 2026) adequately prepare models to balance retrieval breadth and candidate validity, motivating future work on how post-training shapes abstract-search behaviour.

AI Use Statement

Generative AI is part of the methods this paper describes. We use it to generate Open Discovery queries through the Generator–Solver pipeline of Section 3, and we manually review each query and its criteria to confirm that the criteria are faithful to the query and verifiable on the Web. In Entity Recall, annotators write the queries themselves and iterate against a search-augmented model, which also drafts candidate criteria that the annotator keeps, edits, or removes. Our evaluation rests on a model verifying each returned candidate against those criteria, so we validate it directly: the authors manually annotate a subset of candidate–criterion pairs and measure agreement with the verifier (Appendix E). We also use generative AI to search for related work, to plot figures from data we verify ourselves, and for minimal editing of the text. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics Statement

Entity Recall queries were written by annotators recruited on Prolific, and the study was approved by our institution's research ethics committee. Participants read an information sheet describing the task, the data we collect, and how it is handled, and gave informed consent before starting. The task requires AI tools, which the information sheet stated before consent, inviting anyone who preferred not to use them to skip the study. Participation was voluntary and could be stopped at any point, in which case nothing was recorded, and participants could withdraw for seven days after submitting without affecting payment. Pay ranged from $21 to $23 per hour across rounds. We collected no personal details; Prolific IDs were used only for payment, review, and withdrawal, and were removed after seven days, after which a submission cannot be linked to its author. Annotators were asked to choose publicly documented entities and not to include personal information, and we release only anonymised task data. The dataset also reflects who wrote it: Prolific's worker pool may not be fully representative of search users in general, and although we released rounds targeted at different regions (Appendix B), we cannot correct for the pool's underlying skew. Open Discovery queries are generated from seeds grounded in SearchArena traffic, which carries its own demographic skew (Miroyan et al., 2026). Results should therefore be read as measuring search behaviour on the queries we collected rather than on the full distribution of underspecified search.

Reproducibility Statement

Appendix B documents the Entity Recall annotation task, recruitment, calibration solver, and the filters applied before release, with per-stage counts; Appendix C documents the Open Discovery seeds, capability definitions, Generator–Solver loop, and filters. Appendix D gives the models and inference settings, the search harness, the task prompts, and the definition of every metric we report. The prompts used at each stage are reproduced in the appendix, abridged where length required it. The dataset is included in the supplementary materials. We release the dataset, the annotation platform, the generation and evaluation code, the full prompts, and the evaluation traces, so that the pipeline can be rerun and the human study replicated at a larger scale.

References

Anthropic. Claude opus 5 system card. https://www.anthropic.com/claude-opus-5-system-card, July 2026a. System card. Accessed 24 September 2026. 5

Anthropic. Claude Sonnet 5 System Card. https://anthropic.com/claude-sonnet-5-system-card, 2026b. Accessed 22 September 2026. 5

Jaime Arguello, Adam Ferguson, Emery Fine, Bhaskar Mitra, Hamed Zamani, and Fernando Diaz. Tip of the tongue known-item retrieval: A case study in movie identification. In *Proceedings of the 2021 Conference on Human Information Interaction and Retrieval*, CHIIR '21, pp. 5–14, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380553. doi: 10.1145/3406522.3446021. URL https://doi.org/10.1145/3406522.3446021. 1, 3, 9

Hammad Bashir, Kelly Hong, Patrick Jiang, and Zhiyi Shi. Chroma context-1: Training a self-editing search agent. Technical report, Chroma, March 2026. URL https://trychroma.com/research/context-1. 9

Samarth Bhargav, Georgios Sidiropoulos, and Evangelos Kanoulas. 'it's on the tip of my tongue': A new dataset for known-item retrieval. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22, pp. 48–56, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391320. doi: 10.1145/3488560.3498421. URL https://doi.org/10.1145/3488560.3498421. 9

Michael K. Buckland. On types of search and the allocation of library resources. *Journal of the American Society for Information Science*, 30(3):143–147, 1979. doi: https://doi.org/10.1002/asi.4630300305. URL https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/asi.4630300305. 3

Sky CH-Wang, Darshan Girish Deshpande, Smaranda Muresan, Anand Kannappan, and Rebecca Qian. Browsing lost unformed recollections: A benchmark for tip-of-the-tongue search and reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8317–8331, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.406. URL https://aclanthology.org/2025.acl-long.406/. 1, 3, 9

Aileen Cheng, Alon Jacovi, Amir Globerson, Ben Golan, Charles Kwong, Chris Alberti, Connie Tao, Eyal Ben-David, Gaurav Singh Tomar, Lukas Haas, Yonatan Bitton, Adam Bloniarz, Aijun Bai, Andrew Wang, Anfal Siddiqui, Arturo Bajuelos Castillo, Aviel Atias, Chang Liu, Corey Fry, Daniel Balle, Deepanway Ghosal, Doron Kukliansky, Dror Marcus, Elena Gribovskaya, Eran Ofek, Honglei Zhuang, Itay Laish, Jan Ackermann, Lily Wang, Meg Risdal, Megan Barnes, Michael Fink, Mohamed Amin, Moran Ambar, Natan Potikha, Nikita Gupta, Nitzan Katz, Noam Velan, Ofir Roval, Ori Ram, Polina Zablotskaia, Prathamesh Bang, Priyanka Agrawal, Rakesh Ghiya, Sanjay Ganapathy, Simon Baumgartner, Sofia Erell, Sushant Prakash, Thibault Sellam, Vikram Rao, Xuanhui Wang, Yaroslav Akulov, Yulong Yang, Zhen Yang, Zhixin Lai, Zhongru Wu, Anca Dragan, Avinatan Hassidim, Fernando Pereira, Slav Petrov, Srinivasan Venkatachary, Tulsee Doshi, Yossi Matias, Sasha Goldshtein, and Dipanjan Das. The facts leaderboard: A comprehensive benchmark for large language model factuality, 2025. URL https://arxiv.org/abs/2512.10791. 1, 3, 9

Zheng Chu, Xiao Wang, Jack Hong, Huiming Fan, Yuqi Huang, Yue Yang, Guohai Xu, Shengchao Hu, Dongdong Kuang, Chenxiao Zhao, Cheng Xiang, Ming Liu, Bing Qin, and Xing Yu. Redsearcher: A scalable and cost-efficient framework for long-horizon search agents. *arXiv preprint arXiv:2602.14234*, 2026. URL https://arxiv.org/pdf/2602.14234. 9

Charles L. A. Clarke, Maheedhar Kolla, and Olga Vechtomova. *An Effectiveness Measure for Ambiguous and Underspecified Queries*, pp. 188–199. Springer Berlin Heidelberg, 2009. ISBN 9783642044175. doi: 10.1007/978-3-642-04417-5_17. URL http://dx.doi.org/10.1007/978-3-642-04417-5_17. 9

DeepSeek AI. Deepseek v4 model card. https://fe-static.deepseek.com/chat/transparency/deepseek-V4-model-card-EN.pdf, April 2026. Accessed 24 September 2026. 5

Mingyi Deng, Lijun Huang, Yani Fan, Fanqi Kong, Jiayi Zhang, Fashen Ren, Jinyi Bai, Fuzhen Yang, Dayi Miao, Zhaoyang Yu, Yifan Wu, Yanfei Zhang, Fengwei Teng, Yingjia Wan, Song Hu, Yude Li, Xin Jin, Conghao Hu, Haoyu Li, Qirui Fu, Tai Zhong, Xinyu Wang, Xiangru Tang, Nan Tang, Chenglin Wu, and Yuyu Luo. Interactcomp: Evaluating search agents with ambiguous queries. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=AiawINF2T0. 1, 9

George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971, 1987. 1, 3, 9

Google DeepMind. Gemini 3.7 flash. https://deepmind.google/models/model-cards/gemini-3-7-flash/, August 2026. Model card. Accessed 24 September 2026. 5

Nikita Gupta, Riju Chatterjee, Lukas Haas, Connie Tao, Andrew Wang, Chang Liu, Hidekazu Oiwa, Elena Gribovskaya, Jan Ackermann, John Blitzer, Sasha Goldshtein, and Dipanjan Das. Deepsearchqa: Bridging the comprehensiveness gap for deep research agents, 2026. URL https://arxiv.org/abs/2601.20975. 1, 3, 9

Mohammad Javad Hosseini, Filip Radlinski, Silvia Pareti, and Annie Louis. Resolving indirect referring expressions for entity selection. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12313–12335, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.688. URL https://aclanthology.org/2023.acl-long.688/. 3

Deqiang Huang, Jingbo Zhou, Xinjiang Lu, Tong Xu, Hua Wu, and Enhong Chen. Clarify-then-search: A clarification benchmark for deep search with end-to-end nugget restoration. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, pp. 9113–9124. ACM, August 2026. doi: 10.1145/3770855.3817586. URL http://dx.doi.org/10.1145/3770855.3817586. 1, 9

Liwei Jiang, Yuanjun Chai, Margaret Li, Mickel Liu, Raymond Fok, Nouha Dziri, Yulia Tsvetkov, Maarten Sap, and Yejin Choi. Artificial hivemind: The open-ended homogeneity of language models (and beyond). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2026. URL https://openreview.net/forum?id=saDOrrnNTz. 3

Kimiya Keyvan and Jimmy Xiangji Huang. How to approach ambiguous queries in conversational search: A survey of techniques, approaches, tools, and challenges. *ACM Comput. Surv.*, 55(6), December 2022. ISSN 0360-0300. doi: 10.1145/3534965. URL https://doi.org/10.1145/3534965. 1

Mohammad Aflah Khan, Mahsa Amani, Soumi Das, Bishwamittra Ghosh, Qinyuan Wu, Krishna P. Gummadi, Manish Gupta, and Abhilasha Ravichander. In agents we trust, but who do agents trust? latent source preferences steer LLM generations. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=yTUNl6jYGU. 1

Jin Ha Lee, Allen Renear, and Linda C. Smith. Known-item search: Variations on a concept. *Proceedings of the American Society for Information Science and Technology*, 43(1):1–17, 2006. doi: https://doi.org/10.1002/meet.14504301126. URL https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/meet.14504301126. 3

Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Yida Zhao, Liwen Zhang, Litu Ou, Ding-Chu Zhang, Xixi Wu, Xinmiao Yu, Jialong Wu, Xinyu Wang, Zile Qiao, Zhen Zhang, Yong Jiang, Pengjun Xie, Fei Huang, Zhi-Qin John Xu, Shuai Wang, Minhao Cheng, and Jingren Zhou. Websailor-v2: Bridging the chasm to proprietary agents via synthetic data and scalable reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=HuP16O5SJf. 9

Chaitanya Malaviya, Peter Shaw, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. QUEST: A retrieval dataset of entity-seeking queries with implicit set operations. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14032–14047, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.784. URL https://aclanthology.org/2023.acl-long.784/. 1, 9

Chaitanya Malaviya, Joseph Chee Chang, Dan Roth, Mohit Iyyer, Mark Yatskar, and Kyle Lo. Contextualized evaluations: Judging language model responses to underspecified queries. *Transactions of the Association for Computational Linguistics*, 13:878–900, 2025. doi: 10.1162/tacl.a.24. URL https://aclanthology.org/2025.tacl-1.41/. 1, 3

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. AmbigQA: Answering ambiguous open-domain questions. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5783–5797, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.466. URL https://aclanthology.org/2020.emnlp-main.466/. 9

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12076–12100, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.741. URL https://aclanthology.org/2023.emnlp-main.741/. 4

Mihran Miroyan, Tsung-Han Wu, Logan King, Tianle Li, Jiayi Pan, Xinyan Hu, Wei-Lin Chiang, Anastasios Nikolas Angelopoulos, Trevor Darrell, Narges Norouzi, and Joseph E. Gonzalez. Search arena: Analyzing search-augmented LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=MMGRlDnhtI. 1, 5, 10, 16

OpenAI. GPT-5.6 System Card. https://deploymentsafety.openai.com/gpt-5-6, 2026. Accessed 22 September 2026. 5

Thinh Pham, Nguyen Phan Nguyen, Pratibha Zunjare, Weiyuan Chen, Yu-Min Tseng, and Tu Vu. SealQA: Raising the bar for reasoning in search-augmented language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=zWb7ueH16c. 3

Qwen Team. Qwen3.8-max: A new bar for coding and cowork, August 2026. URL https://qwen.ai/blog?id=qwen3.8. 5

Shauli Ravfogel, Valentina Pyatkin, Amir David Nissan Cohen, Avshalom Manevich, and Yoav Goldberg. Description-based text similarity. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=W8Rv1jVycX. 3

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2503.05592. 9

Yixiao Song, Yekyung Kim, and Mohit Iyyer. VeriScore: Evaluating the factuality of verifiable claims in long-form text generation. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 9447–9474, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.552. URL https://aclanthology.org/2024.findings-emnlp.552/. 4

Karen Spärck-Jones, Stephen E. Robertson, and Mark Sanderson. Ambiguous requests: implications for retrieval tests, systems and theories. *SIGIR Forum*, 41(2):8–17, December 2007. ISSN 0163-5840. doi: 10.1145/1328964.1328965. URL https://doi.org/10.1145/1328964.1328965. 1, 9

Yxiling Tao, Shihan Deng, Meiling Tao, Pengzhi Wei, Zhichao Hu, and Zhihao Zhu. When search agents should ask: Discobench for clarification-aware deep search, 2026. URL https://arxiv.org/abs/2606.27669. 1, 9

Diane Tchuindjo, Devavrat Shah, and Omar Khattab. Obliq-bench: Exposing overlooked bottlenecks in modern retrievers with latent and implicit queries, 2026. URL https://arxiv.org/abs/2605.06235. 3

Miriam Wanner, Seth Ebner, Zhengping Jiang, Mark Dredze, and Benjamin Van Durme. A closer look at claim decomposition. In Danushka Bollegala and Vered Shwartz (eds.), *Proceedings of*

*the 13th Joint Conference on Lexical and Computational Semantics (*SEM 2024)*, pp. 153–175, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.starsem-1.13. URL https://aclanthology.org/2024.starsem-1.13/. 4

Jason Wei. Asymmetry of verification and verifier's rule, 2026. URL https://www.jasonwei.net/blog/asymmetry-of-verification-and-verifiers-law. 4

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. Measuring short-form factuality in large language models, 2024a. URL https://arxiv.org/abs/2411.04368. 3

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents, 2025. URL https://arxiv.org/abs/2504.12516. 1, 3, 9

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Zixia Hu, Jie Huang, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, and Quoc V Le. Long-form factuality in large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=4M9f8VMt2C. 6, 30, 34

Ryan Wong, Jiawei Wang, Junjie zhao, Li Chen, Yan Gao, Zhanglong, Xuan Zhou, Zuo Wang, Kai Xiang, Ge Zhang, Wenhao Huang, Yang Wang, and WANG KE. Widesearch: Benchmarking agentic broad info-seeking. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Q7YUY7zGkZ. 1, 3, 9

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259/. 3

Weihao Zeng, Keqing He, Chuqiao Kuang, Xiaoguang Li, and Junxian He. Pushing test-time scaling limits of deep search with asymmetric verification. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=hxL4Uf9tR3. 4

Weinan Zhang, Junwei Liao, Ning Li, Kounianhua Du, and Jianghao Lin. Agentic information retrieval, 2025. URL https://arxiv.org/abs/2410.09713. 1

APPENDIX

A Underspecification in Real Search Queries **16**

B Entity Recall Dataset Construction **20**

C Open Discovery Dataset Construction **21**

D Evaluation Setup **27**

E Verifier Validation **31**

F Results Breakdown **32**

G Calibration Analysis **33**

H Sensitivity to K **34**

I Case Studies: Queries No Model Solved **34**

J Source Analysis **39**

K Diversity of Candidate Solutions Across and Within Models **39**

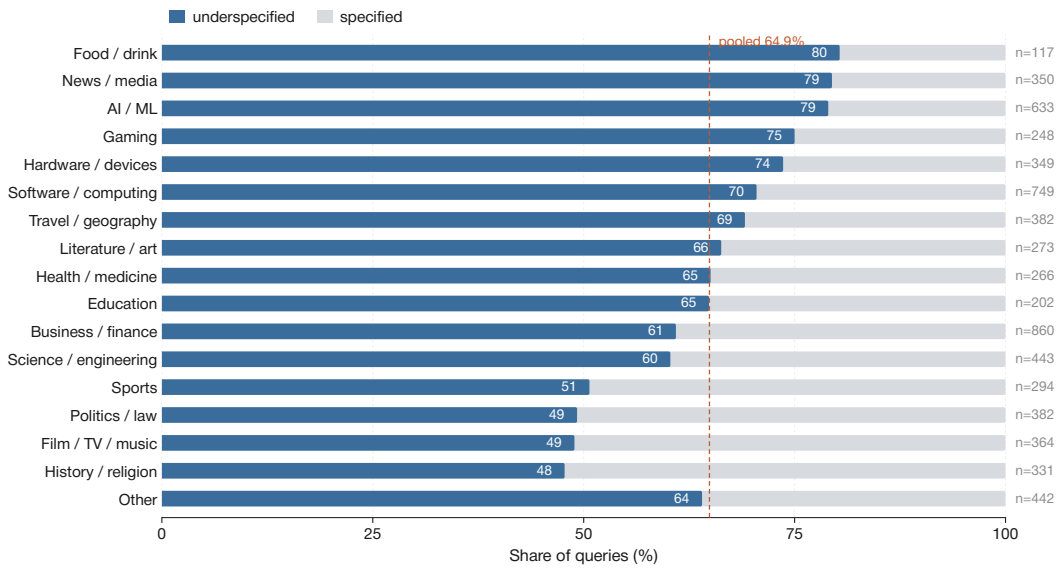


Figure 5: **Most SearchArena retrieval queries are underspecified across domains.** Bars show each domain's share of underspecified queries; the dashed line marks the pooled 64.9%, and n gives the domain's query count. *Other* collects unmatched domain labels and appears last.

A Underspecification in Real Search Queries

We estimate how often real search queries are underspecified using SearchArena (Miroyan et al., 2026), a public corpus of 24,069 anonymised search conversations. We apply two filters. First, we keep records whose language annotation is English only, leaving 12,849 conversations; we drop mixed-language records because the specification judgement depends on close reading of the query. Second, we keep the three intent labels that describe requests to retrieve information about the world (Factual Lookup, Info Synthesis, and Recommendation) and discard the other six (Analysis, Explanation, Guidance, Creative Generation, Text Processing, and Other). This leaves 6,685 queries. For each conversation, we use the first user turn. We label each query with GPT-5.4-mini, accessed through OpenRouter[4], using the query text alone. The classifier assigns one of two labels, which operationalise the distinction in Section 2: *specified* (HIGH_SPECIFIED in the prompt), when the query targets a specific entity or fact with a unique or narrowly bounded answer, including targets described indirectly; and *underspecified* (LOW_SPECIFIED), when multiple candidates could satisfy the query, as in open discovery, recommendation requests, and incomplete recall. Figure 6 gives the full prompt, and Table 3 shows example queries with their labels.

The classifier labels 4,339 of the 6,685 queries (64.9%) as underspecified. To check that no single domain drives this result, we assign each query a domain with a separate LLM pass and map the free-text labels onto sixteen domains. The underspecified share exceeds half in thirteen of the sixteen, ranging from 47.7% in History / religion to 80.3% in Food / drink (Figure 5). A second pass (Figure 7) separates two kinds of underspecified query. *Entity-retrieval* queries expect a set of discrete candidates, each checkable against the stated criteria (e.g., "list movies where a character dies in a car crash"); *descriptive* queries expect prose that explains or summarises a topic (e.g., "tell me about the recent election"). Of the underspecified queries, 1,414 (32.6%) are entity retrieval and 2,925 (67.4%) are descriptive. Underspecified entity retrieval, the setting our benchmark targets, therefore accounts for 21.2% of all retained queries.

These figures are approximate. The labels come from a single model, SearchArena users skew towards technical topics (Figure 5), and the Recommendation intent includes subjective requests, which our benchmark excludes. We use the figures only to show that underspecified search is common in practice.

[4] https://openrouter.ai/

Label	Query	Reason
SPECIFIED	*"What is the name of the restaurant at the top of The Shard in London?"*	Direct lookup: one named landmark, one answer.
SPECIFIED	*"What's the Regular Show episode where Mordecai and Rigby prank Benson by ordering a giant sandwich?"*	Described rather than named, but the scene occurs in only one episode.
SPECIFIED	*"Which event won the most memorable event award at Fuorisalone 2025 in Milan?"*	Named event, year, and award: the constraints fix the answer.
UNDERSPEC.	*"List several 1999 films that have scenes in a library or a bookshop"*	Open set: nothing fixes which or how many qualifying films to return.
UNDERSPEC.	*"What's the best nylon guitar for a person on a budget?"*	Recommendation: "best" and "budget" have no single correct value.
UNDERSPEC.	*"reseach the most starred/forked projects on github this MONTH"*	Arbitrary selection: the cut-off is unstated and the ranking changes.

Table 3: **Example SearchArena queries with their specification labels.** Queries are quoted verbatim. Specified queries fix a single answer, even when the target is described rather than named; underspecified queries leave the answer open through an open set, a recommendation, or an arbitrary selection.

Specification classifier prompt

SYSTEM INSTRUCTION

```
You are classifying search queries by their specification level: does the query
point to a specific entity or fact with a verifiable ground truth, or is it
open-ended with multiple valid candidates?

Classify as one of:

HIGH_SPECIFIED — The user is looking for a specific entity, fact, or answer.
A unique or narrowly bounded ground truth exists.  The query provides enough
constraints that a correct answer can in principle be verified. This includes:
   - Direct lookups ("What year did X happen?")
   - Named-entity fact retrieval ("What are the side effects of drug X?")
   - Constrained retrievals where only one entity satisfies all conditions, even
if described indirectly
   - Yes/no or confirmation queries about a specific claim

LOW_SPECIFIED — The user is not looking for a single answer. Multiple candidates
could satisfy the query and there is no verifiable ground truth. This includes:
   - Open discovery and exploratory queries
   - Recommendation requests
   - Vague recall searches where the description is incomplete enough that
multiple candidates are acceptable

Output JSON only, no markdown, no backticks:
   {"label":  "HIGH_SPECIFIED"  |  "LOW_SPECIFIED",  "confidence":  <0.0-1.0>,
"reasoning": "<one or two sentences>"}
```

USER PROMPT

```
Classify this search query:

<query>
{}
</query>
```

Figure 6: **Specification classifier prompt.** GPT-5.4-mini labels each SearchArena query as specified or underspecified from the query text alone.

Underspecified Subtype Classifier Prompt

SYSTEM INSTRUCTION

You are classifying search queries that are already known to be low-specified (i.e. they do not point to a unique ground truth). Your task is to distinguish between two subtypes:

ENTITY_RETRIEVAL – The query expects a set of discrete, nameable entities as output. The user has explicit criteria that each returned candidate must satisfy, making individual results verifiable. A checker could in principle confirm whether each candidate meets the stated conditions.
Examples:
- "List several 1999 films that have scenes in a library or a bookshop"
- "Which countries have laws that allow you to give up ownership of land outside your own country"
- "Scientific studies using Underpotential Deposition of Mercury for examining rhodium-containing alloys"
- "Which US states rely the most on exports to China"

DESCRIPTIVE – The query expects qualitative information, explanation, or summary. There are no discrete entities to retrieve and verify – the output is prose that describes, explains, or summarises a topic.
Examples:
- "How has the subsidy for heat pumps been reduced in France, Germany, and Britain"
- "Tell me all about clonazepam"
- "What is the current state of China's economy"

Output JSON only, no markdown, no backticks:
{"label": "ENTITY_RETRIEVAL" | "DESCRIPTIVE", "confidence": <0.0-1.0>, "reasoning": "<one or two sentences>"}

USER PROMPT

Classify this search query:

<query>
{}
</query>

Figure 7: **Underspecified Subtype Classifier Prompt.** GPT-5.4-mini separates underspecified queries into entity-retrieval and descriptive subtypes.

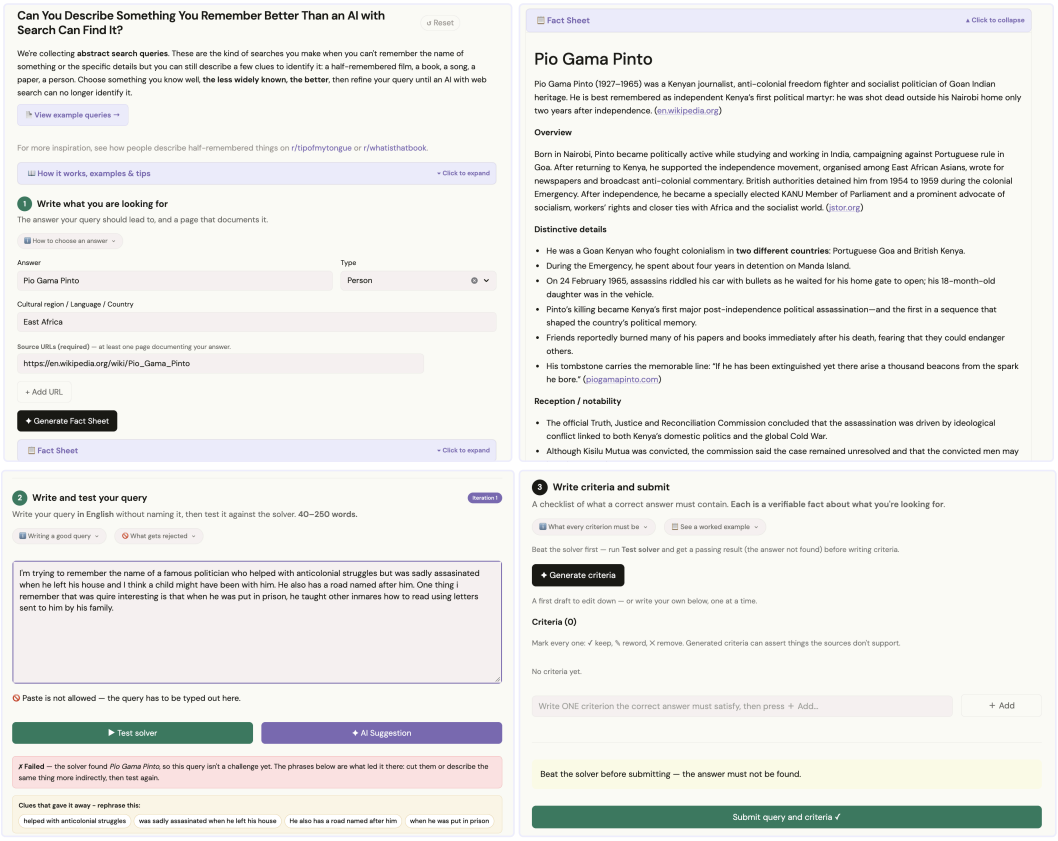


Figure 8: **Entity Recall annotation platform.** Web interface used by annotators to construct, test, and refine queries before submission.

B Entity Recall Dataset Construction

Annotation task. Annotators write each query in a browser tool shown in Figure 8 that we build for this task. We ask them to choose an entity they know well, preferably a cultural or language specific entity and record its type, region, and at least one source URL; the tool then gathers web evidence about it into a fact sheet. They write an English query describing the entity without naming it, test it against the calibration solver, and revise and re-test until the solver fails to retrieve the entity. They then review drafted grading criteria, keeping, editing, or removing each, and submit. The tool records every intermediate version of the query, so the edit history forms part of the released data. A submission takes 15 to 25 minutes, most of it spent waiting on the solver, which takes one to three minutes per run.

Recruitment and consent. This study was approved by our institution's research ethics committee. We recruited annotators on Prolific, restricted to annotators who had passed the AI Task Asessment and are fluent in English, in rounds targeted at different regions to widen the geographic spread of entities. Pay ranged from $21 to $23 per hour across rounds. Participants read an information sheet and gave informed consent before any form was shown. Participation was voluntary and could be stopped at any point, in which case nothing was recorded. Participants could withdraw for seven days after submitting by emailing their Prolific ID, after which the submission was deleted in full without affecting payment. Prolific IDs were used only for payment and withdrawal and were removed after seven days.

Calibration solver. A query is accepted only once the calibration solver has failed to retrieve the entity the user had in mind. The solver has native web search and is run at a fixed budget of medium search context and medium reasoning effort. Four solvers were used across the collection rounds, and the one an annotator faced is fixed for the whole of a submission: GPT-5.4 Mini (82 released

Stage	Removed	Remaining
Prolific submissions	–	352
Parametric-recall screen	45	307
Faithfulness / near-duplicates	5	302
Final author review	47	255

Table 4: **Entity Recall filtering funnel.** Number of queries removed and retained at each stage.

queries), GPT-5.6 Luna (74), GPT-5.4 (64) and GPT-5.6 Sol (35). The solver is never named or shown in the interface, and annotators may re-test as often as they like. Which solver a query was calibrated against is released with it, and Appendix F reports how much harder the stronger baselines make a query.

Filtering. Table 4 shows the filtering pipeline. We begin with 352 Prolific submissions. For the parametric-recall screen, GPT-5.4 answers each query without tools across five samples; we remove a query if any sample recovers the target entity. This removes 45 queries, leaving 307. Faithfulness and near-duplicate checks remove a further 5, leaving 302. Finally, the authors manually assess each query for specification by asking whether the stated constraints, taken together, uniquely identify the seed entity. Queries judged to uniquely determine the seed are removed. This is a manual task-validity check based on the query as written, rather than an exhaustive enumeration of all possible candidates on the Web, removing 47 and leaving 255 released queries.

Composition. Figures 9 and 10 give the entity-type and geographic distributions of the released queries, as recorded by annotators at submission.

Edit-type annotation. GPT-5.5 compares each query version with the one before it and assigns one or more edit types to the rewrite, constrained to JSON output. The unit is one record per rewrite, so a query with four turns yields three records. The prompt instructs the model to record rather than judge: it does not assess query quality or try to identify the entity, every label must be supported by a quote appearing verbatim in one of the two turns, and a rewrite that switches to a different entity is marked and excluded rather than labelled. Labels are not mutually exclusive, and most rewrites take one or two. Three rules resolve the common overlaps: information made indirect counts as ABSTRACTS_DETAIL rather than a drop, since the fact survives; DROPS_NAME and WIDENS_TIME take priority over DROPS_DETAIL, which is reserved for what is neither a name, a place, nor a time; and RESTARTS and PARAPHRASES are each exclusive. Figure 12 gives the prompt, and Figure 11 shows how often each type occurs and how often the frequent types co-occur on the same rewrite.

C OPEN DISCOVERY DATASET CONSTRUCTION

We generate Open Discovery queries with a Generator–Solver loop, then pass them through automated filters and author review.

Input seeds. Each generation begins with a seed specifying a domain, a list of entity types, and a target search capability. Domains and entity types are grounded in real-world search behaviour in SearchArena (Appendix A), giving 12 domains and 106 entity types (Table 5). We shuffle seeds and assign them round-robin, so no combination is favoured. The Generator also sees up to 40 earlier queries from the same run and is instructed not to repeat them to ensure diversity in the style.

Search capabilities. Table 6 defines the four capabilities by the verification each requires, with a released query and its criteria. The Generator receives an extended version of its assigned definition with two worked examples.

Generator–Solver loop. Given a seed, the Generator searches the Web and writes a query with the criteria needed to verify candidates. A solver model then tries to find a shortcut, such as a ready-made list or a single page that answers the query. If it finds one, the evidence returns to the Generator, which revises the query. The loop runs for at least 3 and at most 10 rounds and stops once no shortcut is found. All agents run on GPT-5.4 or GPT-5.4-mini with unrestricted Web access. Released queries took a mean of 3.5 rounds. Figures 14 and 15 give the prompts.

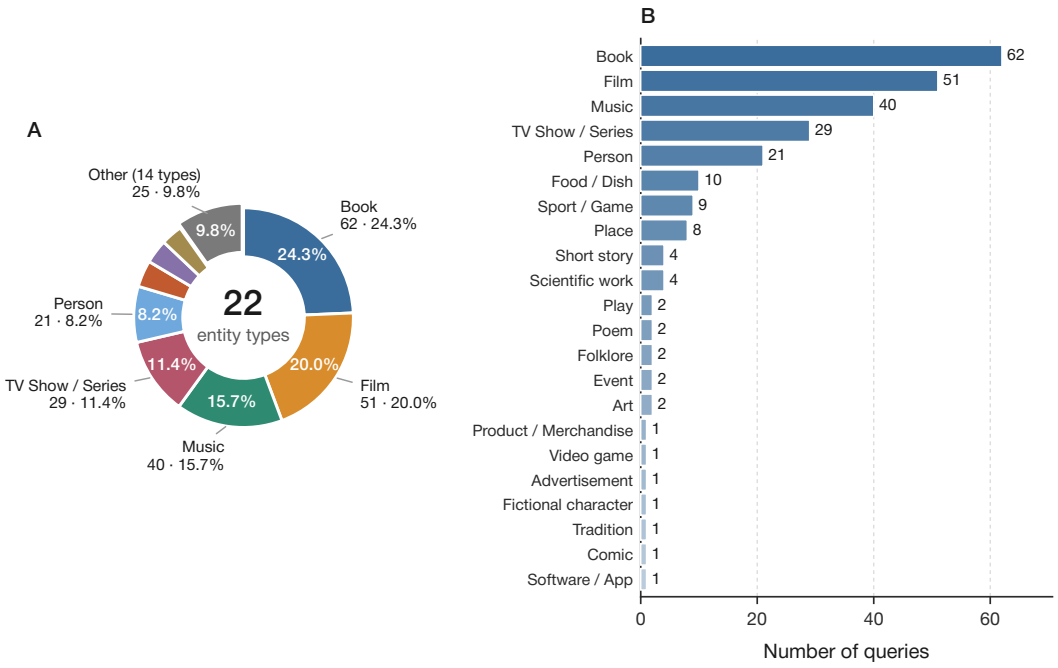


Figure 9: **Entity types in Entity Recall.** Distribution of the entity types annotators chose.

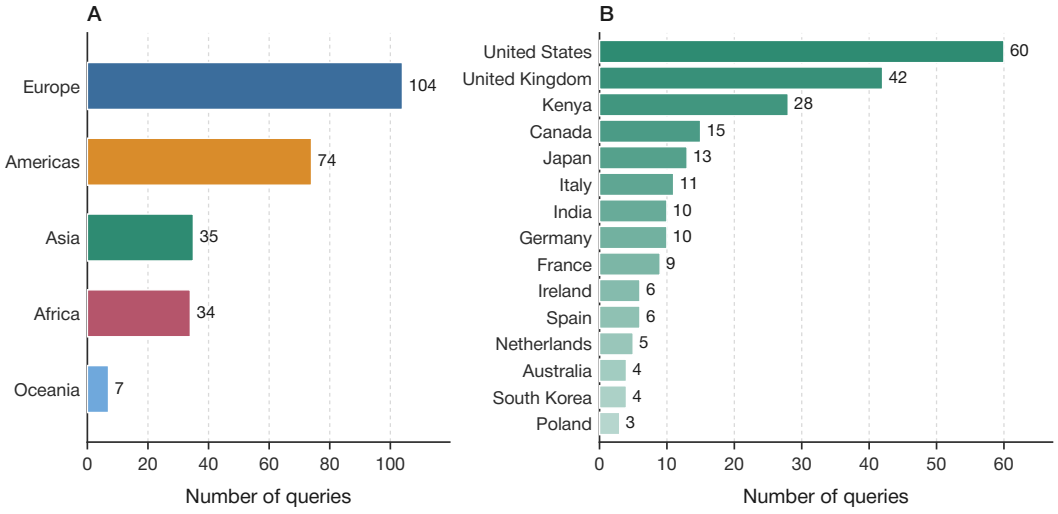


Figure 10: **Geographic distribution of entities.** Cultural region of each entity, as recorded by the annotator.

Filtering and review. Queries then pass through automated filters and review (Table 7). The filters remove queries that are subjective or unverifiable, that collapse to a single answer, or that no longer require their target capability, and repair criteria that do not faithfully reflect the query. The authors then review each remaining query, removing those that pass every filter but are not genuinely open-ended, and checking that its criteria are faithful to the query and verifiable on the Web. Of 1,039 generated queries, 260 are released.

Dataset statistics. Figure 13 shows the domain and entity-type distributions of the released queries. Queries have a median length of 27 words and carry between 3 and 10 criteria (mean 6.05). Each query underwent a mean number of 3.5 Generator–Solver iteration turns.

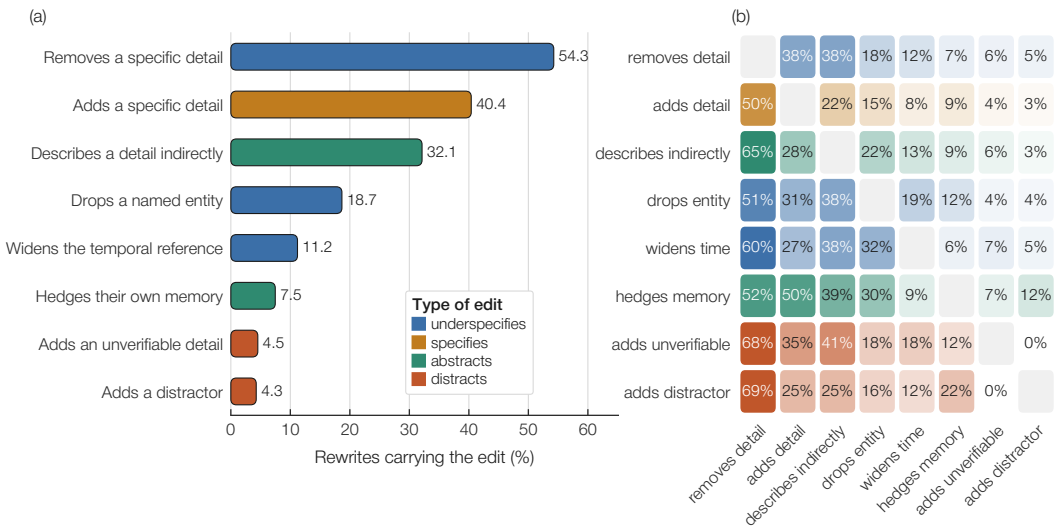


Figure 11: **What annotators change between query versions.** Frequency of each edit type, with the co-occurrence of the frequent types on the same rewrite.

Domain	Types	Entity types (first four)
Science and technology	9	Research papers, Research labs, Scientific instruments, Open-source software projects
Finance and business	10	Public companies, Startups, IPOs, Mergers and acquisitions
Politics and policy	9	Bills and legislation, Elections, Politicians, Government agencies
History and biography	9	Historical figures, Historical events, Battles, Monuments
Film and television	9	Films, TV series, Directors, Production studios
Music	9	Albums, Songs, Artists and bands, Record labels
Literature and writing	7	Books, Authors, Literary awards, Editions and translations
Sports	10	Clubs, Players, Seasons, Leagues
Video games and gaming	8	Games, Studios, Publishers, Consoles
Health and medicine	10	Drugs, Clinical trials, Medical devices, Diseases
Education and learning	7	Universities, Degree programmes, Research centres, Scholarships
Geography and travel	9	Cities and towns, Transport routes, Stations and airports, National parks

Table 5: **Seed domains and entity types.** Each seed pairs one domain with a target capability and a shuffled list of that domain's entity types. The table shows the number of entity types per domain and the first four of each.

Edit-type annotation prompt (abridged)

SYSTEM INSTRUCTION

You annotate how a person rewrote a search query across successive attempts.

Background: a contributor is writing a hard question from memory about something real. After each attempt an automated solver tries to identify what they mean. The contributor sees the solver's guesses and rewrites to make it fail. Your job is to record what kind of change each rewrite made.

Record, do not judge. Do not assess query quality. Do not try to work out the answer.

Unit of annotation. One record per rewrite. Turn 1 has no predecessor, so a query with four turns gives three records. Compare each turn against the one immediately before it.

Labels. Apply every label that fits. Most rewrites take one or two.

1. DROPS_NAME – a named anchor disappears or becomes a description: people, titles, brands, organisations, and also countries, cities and nationalities.
2. WIDENS_TIME – a date, year, decade or age is removed or widened to a span that still contains the answer.
3. DROPS_DETAIL – a concrete fact that is not a name, place or time goes, and nothing replaces it.
4. ABSTRACTS_DETAIL – the fact survives but is expressed indirectly, so a reader who knows the answer still recognises it while a keyword search would not.
5. HEDGES_MEMORY – doubt about the recollection is added or strengthened; content stays, commitment to it weakens.
6. ADDS_DISTRACTOR – new material resembles a different, usually more famous thing and would pull a solver toward it.
7. ADDS_UNRECORDED – new material probably true to the person but unlikely to be written down anywhere a search engine can reach.
8. ADDS_SIGNAL – new material that genuinely narrows toward the answer, or an earlier cue coming back.
9. RESTARTS – the description is rebuilt rather than edited; almost no wording survives, though the thing described is the same.
10. PARAPHRASES – wording, spelling, ordering, or the request made of the solver changes, and the information content does not.

Per record, give turn_from, turn_to, labels, change_count, evidence (one short verbatim quote per label, taken from the before or after text), target_switch, and an optional one-sentence note. Every evidence quote must appear verbatim in one of the two turns it relates to.

Output JSON only. No commentary, no code fences:

```
{"query_id": "string", "records": [{"turn_from": 1, "turn_to": 2,
  "labels": ["DROPS_NAME", "ABSTRACTS_DETAIL"], "change_count": 4,
  "evidence": {"DROPS_NAME": "<quote>"}, "target_switch": false,
  "note": ""}]}
```

USER PROMPT

```
QUERY_ID: {}
TURN 1: <text>
SOLVER 1: correct=<true|false> guess=<name> confidence=<float>
TURN 2: <text>
SOLVER 2: ...
```

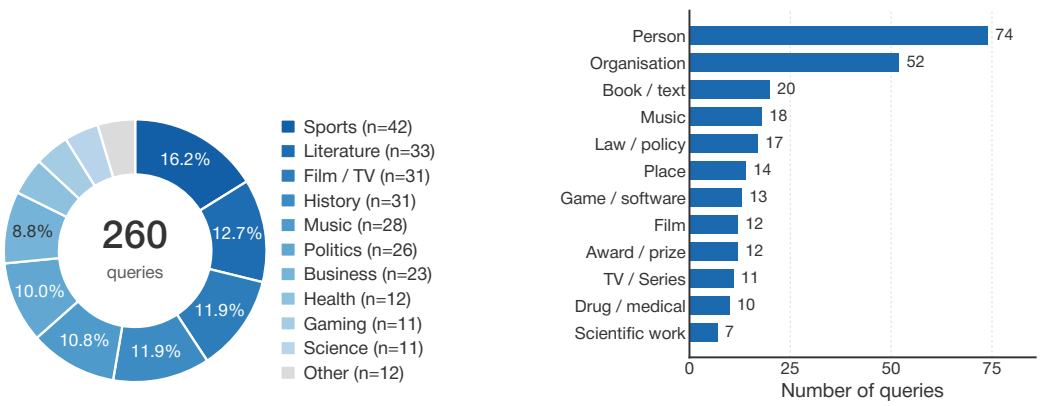
Figure 12: **Edit-type annotation prompt.** GPT-5.5 labels each rewrite between consecutive query versions. Abridged: the full prompt also carries a worked example with a borderline case.

Capability	Definition, example query, and criteria
Cross-domain	The defining property spans two information ecosystems that share no common index, identifier, or vocabulary, so no single source lists both sides. The solver must retrieve each side on its own and join them per candidate. Both sides must be positive, public facts, never a perception. *"Which football players were first capped by a senior national team before later working as hospital nurses?"* • The candidate is a football player. • The candidate was capped by a senior national team. • The candidate later worked as a hospital nurse. • The nursing work began after the first senior national team cap. • The player and the nurse are the same named person.
Negative evidence	Membership is defined by something that did not happen, over a bounded, enumerable universe where the absence is checkable against a record. The solver must enumerate the universe and confirm non-occurrence per candidate. The universe must be closed; open-world negatives are unverifiable and banned. *"which WHO routine vaccines have never, across all WHO position paper revisions since first inclusion, had any booster, revaccination, or additional-dose recommendation"* • The candidate is a WHO routine vaccine. • The candidate vaccine has a WHO position-paper history that can be examined from the vaccine's first inclusion in a WHO position paper through all later revisions of that position paper. • The candidate has no booster recommendation in any WHO position paper revision. • The candidate has no revaccination recommendation in any WHO position paper revision. • The candidate has no additional-dose recommendation in any WHO position paper revision.
Comparative	Membership depends on a candidate's standing relative to a peer group on a measurable axis that no source pre-computes. The solver must gather the peers, compute the statistic, then filter. The query must stay set-seeking and never collapse to a single winner. The axis is a public number. *"which world heritage sites had around 3 million visitors in 2012 but weren't in the top 10 most visited that year?"* • The candidate is a UNESCO World Heritage site. • The candidate recorded roughly 3 million visitors in 2012 (approximately 2.5–3.5 million), according to a published visitor figure for that year. • The candidate appears on a published 2012 ranking of the most-visited World Heritage sites. • The candidate's position on that ranking is 11th or lower (not in the top 10). • The top-10 boundary in criterion 4 and the candidate's position both come from the same ranking named in criterion 3.
Nested sequence	Verifying a candidate requires first resolving a dependent fact, or tracing an ordered multi-step sequence where the links and their order are the point, not two parallel facts joined by "and". The sequence must be documented and ordered. If there is causation, a source must state it. *"i'm looking for films whose lost print was rediscovered, then used for a new restoration, and whose restored version later had its first screening at a named film festival"* • The candidate is a film. • The candidate had a lost print rediscovered. • The rediscovered print was used for a new restoration. • The restored version later had its first screening at a named film festival. • The film's restored version had its first screening at a named film festival after the restoration was completed.

Table 6: **The four target search capabilities.** Each definition is the text given to the Generator, followed by a released query and the criteria used to verify its candidates.

Stage	What it removes or repairs	Removed	Repaired
Automated filters			
Subjectivity and verifiability	Subjective, or not verifiable on the open Web	248	–
Target capability	Target capability absent or no longer necessary	179	–
Specified answer	Singular ask, or criteria converge on one entity	8	–
Criteria fidelity	Criteria not faithful to the query	–	293
Author review			
Manual review	Queries that are not genuinely open-ended, or whose criteria are unfaithful or unverifiable	211	–
Generated		906	
Released		260	

Table 7: **Automated filters and author review remove most generated queries.** Records removed or repaired at each stage. The target-capability filter runs both inside the Generator–Solver loop and after it. Automated filters run on GPT-5.4 without tools; filter prompts are in the code release.



(a) Domain

(b) Entity type

Figure 13: Distribution of the 260 released queries by (a) domain, with domains under 10 queries folded into *Other*, and (b) entity type.

Model	Identifier	Effort	Structured output
GPT-5.6 Sol	`openai/gpt-5.6-sol`	high	Yes
Claude Opus 5	`anthropic/claude-opus-5`	high	Yes
Gemini 3.7 Flash	`google/gemini-3.7-flash`	high	Yes
DeepSeek V4 Pro	`deepseek/deepseek-v4-pro-0813`	high	No
GPT-5.6 Luna	`openai/gpt-5.6-luna`	high	Yes
Claude Sonnet 5	`anthropic/claude-sonnet-5`	high	Yes
Qwen 3.8 Max	`qwen/qwen3.8-max`	xhigh	No
Verifier models			
Criterion verifier	`google/gemini-3.7-flash`	medium	Yes
Hit Rate judge	`gpt-5.6-luna`	default	No

Table 8: **Models and inference settings.** Identifiers are OpenRouter model names, except the Hit Rate judge, which we call through the OpenAI API without tools.

D Evaluation Setup

D.1 Models

Table 8 lists the evaluated and verifier models. We run every evaluated model at reasoning effort `high`, except Qwen 3.8 Max, which runs at `xhigh`. We leave temperature unset. For five models, we constrain the final answer to the candidate schema through structured outputs. For DeepSeek V4 Pro and Qwen 3.8 Max, structured outputs suppress tool calling, so we specify the output format in the prompt and parse the response.

D.2 Search Harness

All models use the same two tools and search backend. The `search` tool queries Exa[5] (`type: auto`) and returns ten results, each with a title, URL, and the first 500 characters of the page. The `read` tool fetches a page through Jina Reader[6] and truncates it to 30,000 characters at a line boundary, telling the model when truncation occurs. The tool descriptions are:

- `search(query)`: "Search the web. Returns the top ranked results, each with a title, URL, and snippet. Pass a natural-language query describing what you are looking for. The number of results is fixed; run another search to see more."
- `read(url, respond_with)`: "Read the text content of a web page with Jina Reader."

Search and read calls are uncapped. Each trajectory is limited to 100 model turns; if a run reaches the limit, we withdraw the tools and ask the model for a final answer. We retry transient failures, such as rate limits, until completion. In the parametric baseline, we provide no tools and reject any tool call, so each query receives exactly one model call.

D.3 Task Prompts

Figures 16, 17, and 18 give the Entity Recall, Open Discovery, and parametric baseline prompts. The definition of confidence is identical across all three.

D.4 Run Termination

A run ends when the model answers, when it reaches the turn limit, or when a request fails. Table 9 reports turn counts and termination outcomes. The limit rarely binds: 22 of 1,781 Entity Recall runs (1.2%) and 31 of 1,820 Open Discovery runs (1.7%) reach it, most from DeepSeek V4 Pro. Four Entity Recall runs failed without producing a trajectory: three exceeded the model's context window (two from GPT-5.6 Luna, one from GPT-5.6 Sol), and one received an empty provider

[5] https://exa.ai/
[6] https://jina.ai/

Generator prompt (abridged)

```
You are writing a single web-search query that a real person would type into a
chat window, then hardening it across several turns against a solver.

Your query must hit four bars at once:
   1. OBJECTIVE - every defining property is a recorded fact, not a judgment or
a perception.
   2. NATURAL - it reads like one thing a real person typed, not a specification.
   3. DIFFICULT - a strong search-augmented model cannot answer it from memory
or from one page; the difficulty comes from the SEARCH CAPABILITY below.
   4. OPEN - the person asking has a SET OF CRITERIA, not one specific thing
they are trying to re-find. The criteria should leave a genuine SET open to
discovery.

YOUR ASSIGNMENT
Domain: {domain}
Entity type: choose ONE from this list - the answer is a SET of real, named
instances of it:
   {entity_type_options}

TARGET SEARCH CAPABILITY - {capability_name}
   {capability_def}

The difficulty of your query must come from THIS capability. Build the query so
that removing the part that forces this capability would make the query trivial.

{prior_queries_block}{turn_info}{feedback_block}

TOOLS
   search(query) - Web search. Returns title, URL, snippet.
   read(url) - Fetches the full text of any URL.

THE LISTICLE TEST. Ask: would a "Top 10" or "List of" article plausibly exist
for this exact category? If yes, it is too easy - a journalist or fan wiki
already did the discovery work and the solver will find that page.

CRITERIA. Write 5-8 criteria a fact-checker uses to verify a candidate: plain
factual claims, each answerable yes/no by looking the candidate up. Criterion
1 always states what the candidate IS. Each later criterion states exactly one
atomic fact and traces to a phrase in the query. No criterion may be about where
a fact is documented.

OUTPUT. Output JSON only:
   {
      "chosen_angle": "the angle you built on and how it uses the capability",
      "capability_is_load_bearing": "what trivialises the query if removed",
      "final_query": "final query text",
      "entity_type": "short noun phrase naming the set",
      "criteria": [{"criterion": "...", "traces_to": "phrase in the final
query"}]
   }
```

Figure 14: **Generator prompt.** Abridged for space: the full prompt additionally specifies query style, banned constructions, and common ways a query can look hard while remaining easy. Placeholders in braces are filled from the seed, earlier queries, and Solver feedback. The complete prompt is in the code release.

Solver prompt (abridged)

```
You are a query solver working with a query generator over several rounds to
produce natural but genuinely challenging open-discovery queries. Each round
you attempt to find the answer to the current query with real web searches, then
report what you retrieved and what, if anything, was easy.

Query: {query}

Criteria that define a valid candidate:
   {criteria}

{history_block}

INTENDED CAPABILITY - {capability_name}
   {capability_def}

The shortcut you are hunting for is a source that has already done this work
for the reader: a page that performed the join, a pre-made list, a single table
that encodes the discriminating clauses.

TWO TESTS - RUN IN ORDER

TEST 1 - PARAMETRIC MEMORY. Before searching: can you name specific candidates
that satisfy ALL the criteria right now, from memory, with high confidence? If
yes, the answer is already common knowledge.

TEST 2 - SOLVE IT. Attempt the query the way a capable search-augmented model
would. Try the exact query, then rewordings that retitle the intent the way an
aggregator would; expand with terms not in the query; open the most promising
results and judge from the page, not the snippet.

WHAT MAKES A QUERY EASY. If the query cracked, name which shortcut did it: a
READY-MADE LIST containing items that satisfy every clause; SURFACE VOCABULARY,
where the query's own words returned the candidates; RECALL, where the answers
are well known; or ONE-PAGE SYNTHESIS, where a single source did the whole join.

A CANDIDATE UNIVERSE IS NOT A READY ANSWER. A list of the base entity does
not encode the query's discriminating clauses. If you would still have to go
candidate-by-candidate to confirm who qualifies, no ready answer surfaced.

VERDICT
   too_easy - You found an easy path and can name it.
   pass - You tried hard and no shortcut handed over the answer.

FEEDBACK. Report which constraint did the work, which was decorative, what was
easy or softest, whether the query's own words matched the corpus, and one
concrete revision that would harden it. Ground every claim in something you
retrieved.

OUTPUT. Output JSON only, with the searches you ran, the candidates you found,
a report carrying the verdict and its reasoning, and feedback to the generator.
```

Figure 15: **Solver prompt.** Abridged for space: the full prompt additionally specifies the difficulty signals a passing query must show and the failure-mode tags used in feedback. The complete prompt is in the code release.

Model	Entity Recall			Open Discovery		
	Median turns	At cap (%)	No answer (%)	Median turns	At cap (%)	No answer (%)
GPT-5.6 Sol	7	0.0	0.0	19	0.4	3.1
Claude Opus 5	5	0.0	2.8	9	0.0	0.4
Gemini 3.7 Flash	8	0.0	0.8	20	0.4	1.9
DeepSeek V4 Pro	26	8.6	1.2	51	9.2	5.8
GPT-5.6 Luna	12	0.0	0.0	29	1.9	7.7
Claude Sonnet 5	4	0.0	3.1	7	0.0	0.0
Qwen 3.8 Max	7	0.0	11.8	18	0.0	3.1
All	8	1.2	2.8	19	1.7	3.1

Table 9: **Run termination by model and task.** Median model turns per run, share of runs reaching the 100-turn limit, and share of runs that ended normally without a parsable candidate list. Entity Recall covers 1,781 runs, since GPT-5.6 Sol and GPT-5.6 Luna each lost two runs to request failures; Open Discovery covers 1,820.

response (GPT-5.6 Sol). Separately, 2.8% of Entity Recall runs and 3.1% of Open Discovery runs ended normally but returned no parsable candidate list. We score failed and unanswered runs as zero on every metric, so all models are evaluated over the same queries.

D.5 METRICS

For a model's response to one query, let n be the number of returned candidates and S is the number of *distinct* candidates satisfying *every* criteria. We compute each metric per query and macro-average across queries.

Precision. The fraction of returned candidates that are valid:

$$\text{Precision} = \frac{S}{n}. \tag{1}$$

Coverage@K. Whether enough valid candidates are returned. Because the complete set of valid candidates is unobservable, we follow Wei et al. (2024b) and treat K as a target number of valid candidates for a useful response:

$$\text{Coverage@}K = \min\left(\frac{S}{K}, 1\right). \tag{2}$$

We set K to the median number of candidates returned across all systems and queries within each setting, giving $K = 5$ for Open Discovery and $K = 4$ for Entity Recall. Appendix H reports results across a range of K values.

F1@K. The harmonic mean of Precision and Coverage@K:

$$\text{F1@}K = \frac{2\,\text{Precision} \cdot \text{Coverage@}K}{\text{Precision} + \text{Coverage@}K}. \tag{3}$$

Each of the three metrics is 0 when its denominator is 0. Because we macro-average F1@K per query, the reported F1@K does not equal the harmonic mean of the reported Precision and Coverage@K.

Hit Rate. Whether the seed entity appears among the returned candidates. Over Q Entity Recall queries:

$$\text{HitRate} = \frac{1}{Q}\sum_{q=1}^{Q} \mathbb{1}\big[\text{the seed entity of query } q \text{ is among its returned candidates}\big]. \tag{4}$$

The indicator comes from the Hit Rate judge (Table 8), which agrees with the verifier's seed-entity verdict on 96% of pairs.

Expected Calibration Error. Each model states a confidence $p_i \in [0, 1]$ for its top candidate on each of N queries. We partition these into $B = 10$ equal-width bins $\mathcal{B}_1, \ldots, \mathcal{B}_B$ and compute

$$\mathrm{ECE} = \sum_{b=1}^{B} \frac{|\mathcal{B}_b|}{N} \left| \overline{p}(\mathcal{B}_b) - \overline{o}(\mathcal{B}_b) \right|, \tag{5}$$

where $\overline{p}(\mathcal{B}_b)$ and $\overline{o}(\mathcal{B}_b)$ are the mean confidence and mean outcome in bin b. In the main text, $o_i = 1$ when the top candidate is the seed entity; Appendix G also reports ECE with $o_i = 1$ when the top candidate is valid. Reliability diagrams show only bins with at least 10 observations.

Queries a model does not answer are scored as 0 on every metric rather than dropped, so all models are evaluated over the same queries.

Entity Recall task prompt

```
You are a search assistant with access to web search and page-reading tools.

The user is trying to recall something from memory using an abstract or
underspecified description. They know what they're looking for and will recognise
it among a set of candidates once they see it, even if they cannot specify all
the constraints and concrete details required to retrieve it.

Your task is to understand the query and retrieve candidates that satisfy what
the user is looking for.

For each candidate, confidence is your calibrated estimate (0 to 100) that it
is correct, not a ranking.

Here is the User Query: "{query}"

Return only valid JSON in this shape:
   {
      "candidates": [
         {
            "candidate":  "the  thing  or  combination  of  things  the  query
describes",
            "confidence": 0-100,
            "key_evidence":  "specific  facts  you  found  that  support  this
candidate"
         }
      ]
   }
```

Figure 16: **Entity Recall Task Prompt.** {query} is replaced with the annotator's query verbatim.

E VERIFIER VALIDATION

We validate the verifier's per-criterion judgements against human annotation. The verifier is Gemini 3.7 Flash with medium reasoning effort, Web search and page-reading tools, and a JSON-constrained final response. Figure 19 shows the prompt.

Sampling. We sample 30 candidates, giving 166 candidate–criterion judgements. The sample is distributed across all seven models, contains at most one candidate per query, and is balanced across verifier score bands. Three authors with expertise in fact-checking and NLP independently annotate every judgement using the same information available to the verifier: the query, seed entity, candidate, retrieved evidence, and criterion text.

Inter-annotator agreement. Across annotator pairs, mean raw agreement is 0.859 and mean Cohen's κ is 0.675, with pairwise agreement ranging from 0.831 to 0.898. Across all three annotators, Fleiss' κ is 0.674 and Krippendorff's α is 0.675. All three annotators agree on 131 of the 166 judgements.

Human–verifier agreement. Across the three annotators, the verifier achieves mean raw agreement of 0.904 and mean Cohen's κ of 0.780; per-annotator agreement is 0.898, 0.952, and 0.861. Treat-

Open Discovery Task Prompt

```
You are a search assistant with access to web search and page-reading tools.

The user has a query describing what they are looking for. Your task is to help
the user find candidates that satisfy what they are looking for.

For each candidate, confidence is your calibrated estimate (0 to 100) that it
is correct, not a ranking.

Here is the User Query: "{query}"

Return only valid JSON in this shape:
   {
      "candidates": [
         {
            "candidate":  "the  thing  or  combination  of  things  the  query
describes",
            "confidence": 0-100,
            "key_evidence":  "specific  facts  you  found  that  support  this
candidate"
         }
      ]
   }
```

Figure 17: **Open Discovery Task Prompt.**

Model	Overall (%)	Calibrated against (%)			
		GPT-5.4 Mini $n = 82$	GPT-5.4 $n = 64$	GPT-5.6 Luna $n = 74$	GPT-5.6 Sol $n = 35$
GPT-5.6 Sol	67.45	73.17	60.94	71.62	57.14
Claude Opus 5	64.71	71.95	60.94	67.57	48.57
Gemini 3.7 Flash	61.18	63.41	51.56	67.57	60.00
DeepSeek V4 Pro	53.33	57.32	46.88	54.05	54.29
GPT-5.6 Luna	47.84	51.22	39.06	56.76	37.14
Claude Sonnet 5	38.04	37.80	32.81	48.65	25.71
Qwen 3.8 Max	36.86	39.02	29.69	43.24	31.43
Mean	52.77	56.27	45.98	58.49	44.90

Table 10: **Queries calibrated against stronger baselines are harder**. Hit Rate on Entity Recall by the baseline model each query was calibrated against. Subset sizes sum to the 255 queries.

ing the verifier and annotators as four raters gives Krippendorff's $\alpha = 0.728$. Against the majority human label, the verifier agrees on 94.6% of judgements ($\kappa = 0.876$); on the 131 unanimous judgements, agreement rises to 98.5%. The verifier therefore agrees with human annotators at least as closely as the annotators agree with one another, with most disagreement concentrated on cases that humans also find ambiguous.

F Results Breakdown

Difficulty by calibration model. Table 10 reports Hit Rate by the calibration model used in annotation. Queries calibrated against GPT-5.6 Sol are hardest (mean 44.90%) and those calibrated against GPT-5.6 Luna easiest (58.49%)

Capability breakdown on Open Discovery. Table 11 gives the per-model figures behind Figure 3. Cross-domain queries yield the lowest F1@5 for every model and the shortest candidate lists (5.65 on average, against 6.86 to 8.12 for the other capabilities).

Model	Cross-domain		Nested seq.		Negative ev.		Comparative		All	
	F1@5 (%)	Cand.	F1@5 (%)	Cand.	F1@5 (%)	Cand.	F1@5 (%)	Cand.	F1@5 (%)	Cand.
GPT-5.6 Sol	48.8	4.92	56.4	6.53	57.8	10.87	64.4	10.42	56.5	8.01
Gemini 3.7 Flash	43.0	4.49	50.4	5.87	54.1	5.11	62.4	6.12	52.0	5.37
Claude Opus 5	44.5	7.31	51.2	8.29	52.5	8.05	55.6	8.08	50.7	7.92
DeepSeek V4 Pro	42.0	3.80	48.7	4.71	46.7	7.82	55.8	5.27	48.0	5.33
GPT-5.6 Luna	42.9	4.49	46.3	5.12	47.3	9.00	51.0	8.00	46.7	6.53
Qwen 3.8 Max	36.6	8.97	45.4	11.38	46.0	10.08	51.8	10.64	44.6	10.25
Claude Sonnet 5	30.2	5.55	38.0	6.15	36.9	5.87	42.0	5.88	36.5	5.86
Queries	71		68		62		59		260	

Table 11: **Cross-domain queries are hardest for every model and elicit the shortest candidate lists**. Per-model F1@5 and mean number of candidates returned, by target search capability, on Open Discovery.

G Calibration Analysis

We provide additional details on the calibration analysis in this section. For each returned candidate, models report a verbalised confidence score from 0 to 100, prompted as their estimate that the candidate is the correct entity the user is trying to recall. On Entity Recall, we evaluate these confidences against two outcomes. The *identity outcome* records whether the highest-confidence candidate matches the seed entity, while the *criterion outcome* records whether it satisfies every query criterion. We use accuracy to refer to the percentage of queries for which the highest-confidence candidate satisfies the corresponding outcome. We report expected calibration error (ECE) over 10 equal-width confidence bins (Appendix D.5). We additionally report the calibration gap, defined as mean stated confidence minus the observed outcome rate; positive values indicate overconfidence.

With and without search. Table 12 compares calibration against the identity outcome with and without Web search. Search improves calibration for six of seven models, but most remain substantially overconfident. The improvement comes primarily from accuracy increasing towards already-high confidence rather than models becoming more cautious. Claude Opus 5 is the exception highlighted in the main text: with search, its mean confidence is 56.3% and its accuracy is 55.3%, yielding an ECE of 7.1% and a calibration gap of only 1.0 point. The remaining models have ECEs between 24.4% and 44.6%.

Against criterion validity. Because Entity Recall queries may admit valid candidates other than the seed, we also score the same confidences against whether the highest-confidence candidate satisfies every query criterion (Table 13 and Figure 20). Overconfidence persists: six of seven models still overstate their confidence, by 5.2 to 25.2 points. The gaps are smaller than against the identity outcome, indicating that part of the overconfidence reflects uncertainty over which valid candidate the user intended. Nevertheless, the pattern remains when correctness is relaxed to criterion satisfaction. Claude Opus 5 is the exception in the opposite direction: against criteria it is underconfident by 16.0 points, and it is the only model whose ECE is worse against criteria (16.1%) than against identity (7.1%).

Confidence separation. Calibration measures whether stated confidence has the correct absolute level, whereas separation measures whether confidence distinguishes correct from incorrect predictions. Figure 21 and Table 14 show that every model separates correct from incorrect highest-confidence candidates well, with AUROC between 0.845 and 0.915. Confidence therefore remains informative about whether a selected candidate is correct even when its absolute value is poorly calibrated. Models achieve this separation differently: GPT-5.6 Sol, GPT-5.6 Luna, and Gemini 3.7 Flash concentrate confidence near the upper end of the scale, whereas Claude Opus 5 and Claude Sonnet 5 use a substantially wider range.

Model	Accuracy (%) ↑		Mean confidence (%)		ECE (%) ↓		Gap (%) → 0	
	w/o Search	w/ Search	w/o Search	w/ Search	w/o Search	w/ Search	w/o Search	w/ Search
Claude Opus 5	30.8	55.5	39.1	56.5	9.3	**7.1**	8.3	**+1.0**
DeepSeek V4 Pro	20.1	42.9	59.3	67.1	39.2	24.3	39.2	24.3
Qwen 3.8 Max	21.6	34.7	52.3	61.2	30.7	26.6	30.7	26.6
Claude Sonnet 5	13.4	29.6	44.8	58.5	31.5	29.9	31.4	28.9
GPT-5.6 Sol	40.0	55.7	80.3	88.9	40.3	33.1	40.3	33.1
Gemini 3.7 Flash	36.5	49.8	82.3	85.4	45.8	35.6	45.8	35.6
GPT-5.6 Luna	26.3	41.5	68.8	86.1	42.5	44.6	42.5	44.6

Table 12: **Search improves calibration for most models, but substantial overconfidence remains.** Calibration of the highest-confidence candidate against the identity outcome, with and without search, ordered by ECE with search. Gap is mean stated confidence minus accuracy, so positive values indicate overconfidence and values closer to zero indicate better calibration. Bold denotes the lowest ECE and smallest absolute gap with search.

Model	Mean conf. (%)	Criterion outcome			Identity outcome		
		Accuracy (%)	Gap (%)	ECE (%) ↓	Accuracy (%)	Gap (%)	ECE (%) ↓
Qwen 3.8 Max	61.2	54.2	7.0	9.0	34.7	26.6	26.6
DeepSeek V4 Pro	67.1	61.9	5.2	10.7	42.9	24.3	24.3
Claude Sonnet 5	58.5	48.6	9.9	13.1	29.6	28.9	29.9
Claude Opus 5	56.5	72.5	−16.0	16.1	55.5	1.0	7.1
Gemini 3.7 Flash	85.4	64.8	20.6	22.8	49.8	35.6	35.6
GPT-5.6 Sol	88.9	72.3	16.5	23.7	55.7	33.1	33.1
GPT-5.6 Luna	86.1	60.9	25.2	26.1	41.5	44.6	44.6

Table 13: **Overconfidence persists when correctness is defined by criterion validity rather than seed identity.** The same stated confidences are scored against both outcomes in the search setting, ordered by ECE against criterion validity. Accuracy is the percentage of queries for which the highest-confidence candidate satisfies the corresponding outcome; gap is mean stated confidence minus accuracy, so positive values indicate overconfidence.

H Sensitivity to K

Figure 22 reports Coverage@K for Entity Recall and F1@K for Open Discovery across $K = 1$ to 8. K sets how many valid candidates count as a sufficient response, so raising it rewards breadth and lowers the return on precision alone. We set K to the median number of candidates returned across all systems and queries similar to Wei et al. (2024b), which fixes it before any ranking is computed.

On Entity Recall, ranking is stable: Claude Opus 5 leads Coverage@K at every K. On Open Discovery, GPT-5.6 Sol leads and Claude Sonnet 5 trails at every K, but the models between them reorder with candidate count. Claude Opus 5 (7.92 candidates on average) moves from fifth at $K = 1$ to third at $K = 8$, and Qwen 3.8 Max (10.25) from sixth to fifth, while DeepSeek V4 Pro (5.33) falls from third to sixth. Varying K makes the precision–coverage trade-off in Section 5 clear: as K increases, models that return more valid candidates are rewarded more, while more selective models fall in relative rank.

I Case Studies: Queries No Model Solved

No model recovers the seed entity on 40 of the 255 Entity Recall queries (15.7%), while all seven recover it on 43 (16.9%). We present five of the 40, chosen because they fail in different ways. Each case gives the query and its criteria verbatim as released, and the highest-confidence candidate each model returned, with the number of criteria the verifier judged satisfied.

Model	Median conf., correct (%)	Median conf., incorrect (%)	Difference (%) ↑	AUROC (%) ↑ (95% CI)
Claude Sonnet 5	85.0	47.0	38.0	91.5 [87.0, 95.0]
Claude Opus 5	86.0	27.0	59.0	89.7 [86.0, 93.0]
GPT-5.6 Sol	99.0	88.0	11.0	88.8 [85.0, 93.0]
Qwen 3.8 Max	88.0	45.0	43.0	88.1 [83.0, 92.0]
Gemini 3.7 Flash	95.0	85.0	10.0	86.9 [83.0, 91.0]
DeepSeek V4 Pro	90.0	55.0	35.0	86.8 [82.0, 91.0]
GPT-5.6 Luna	98.0	85.5	12.5	84.5 [79.0, 89.0]

Table 14: **All models separate correct from incorrect highest-confidence candidates well.** Separation of stated confidence between highest-confidence candidates that do and do not match the seed entity, ordered by AUROC. All values are reported as percentages; higher AUROC indicates better discrimination.

Case 1: Criteria satisfied, entity missed

Seed. *How to Hold a Crocodile* (The Diagram Group, 1981) [source] **Turns.** 4

> *I'm trying to remember the name of a book I read when I was young. It's like a self-help sort of book, but it contains loads of silly facts and useless kind of trivia about how to do things, like things few people would ever actually want to do. It was pretty funny as well, not a serious book as I recall.*

Criteria.

C1. The candidate is a book.
C2. The candidate is presented in the style of a self-help or instructional manual.
C3. The candidate is a collection of how-to material rather than a single continuous narrative.
C4. The candidate includes trivia or fact-style entries about how to do things.
C5. The candidate gives instructions for unusual or improbable activities rather than only ordinary practical tasks.
C6. The candidate is intended as a humorous work rather than a serious practical guide.

Model responses.

Model	Candidates	Top candidate	Criteria met
Claude Opus 5	14	*How To* (Randall Munroe)	6/6
GPT-5.6 Sol	11	*The Encyclopedia of Immaturity*	6/6
DeepSeek V4 Pro	6	*How To* (Randall Munroe)	6/6
Gemini 3.7 Flash	6	*The Worst-Case Scenario Survival Handbook*	6/6
Claude Sonnet 5	5	*The Worst-Case Scenario Survival Handbook*	6/6
Qwen 3.8 Max	5	*How To* (Randall Munroe)	6/6
GPT-5.6 Luna	4	*How To* (Randall Munroe)	6/6

Why it fails. Every model returns candidates satisfying all six criteria, and several return five or more, yet none returns the seed. The criteria describe a genre with many members, and the query does not separate the intended one from the rest, which is why criterion satisfaction and entity recovery are measured separately.

Case 2: Regional work, global analogue

Seed. *Kidagaa Kimemwozea* (Ken Walibora), a Swahili novel [source] **Turns.** 1

> *I read or maybe watched something a while back, I'm not really sure what it was. These two people leave home to find a better life somewhere. On the way there's this thing that everyone is scared of but they do it anyway and it kind of changes everything. They end up working for these people who have all the power and money. One of them is supposed to be really respected, like an intellectual, but turns out to be a fraud. The main character has this secret, something to do with a family member who got wronged. What really stuck with me is that at some point the powerful man's own child walks away from all of it and chooses to live with nothing. There was also this person everyone dismissed as mad who turned out to be the only one brave enough to say the truth out loud.*

Criteria.

C1. The candidate is a book
C2. The candidate has two people who leave home to find a better life somewhere
C3. The candidate has those two people do something on the way that everyone is scared of doing
C4. The candidate has those two people end up working for people who have all the power and money
C5. The candidate has one of those powerful and wealthy people who is supposed to be respected as an intellectual but turns out to be a fraud
C6. The candidate has a main character with a secret involving a family member who was wronged
C7. The candidate has a powerful man's own child walk away from that privilege and choose to live with nothing
C8. The candidate has a person everyone dismissed as mad who is the only one to say the truth out loud

Model responses.

Model	Candidates	Best candidate	Criteria met
Gemini 3.7 Flash	2	*Six of Crows* (Leigh Bardugo)	4/8
Claude Opus 5	7	*Buddha* (Osamu Tezuka)	3/8
Claude Sonnet 5	5	*Behold the Dreamers* (Imbolo Mbue)	3/8
DeepSeek V4 Pro	5	*The Thief Lord* (Cornelia Funke)	3/8
GPT-5.6 Luna	1	*Noli Me Tangere* (José Rizal)	3/8
GPT-5.6 Sol	8	*Noli Me Tángere* (José Rizal)	3/8
Qwen 3.8 Max	3	*Noli Me Tángere* (José Rizal)	3/8

Why it fails. The description is specific and its elements are unusual in combination, but models return canonical works with similar themes, such as *Noli Me Tángere*, *The Count of Monte Cristo*, and *Pachinko*, rather than a Swahili novel. No candidate satisfies more than four of the eight criteria.

Case 3: Regional work, wrong genre

Seed. *Salamu Kutoka Kuzimu*, a Swahili spy thriller [source] **Turns.** 1

> *This has been in my head for years and I can't figure out where I know it from, it might have been a book, could have been a film, maybe even a series, I honestly can't remember anymore. There's this group of powerful people who meet somewhere hidden and they're planning something huge, they want to take out important people across an entire continent all at once. But there's this one person who keeps getting in their way and they need to deal with him first. The part I can never shake is how they do it they find this woman who's carrying this horrible secret, she killed someone and hid the body somewhere disgusting, and they hold that over her to make her do what they want. They rough her up badly and basically throw her in front of the guy they want dead, all messed up and bleeding, because they know he's the kind of person who won't walk past someone in pain. Her suffering is literally the trap. I think there was something about making his death look like it was natural, like a disease or something. This has been driving me crazy, does anyone recognise this?*

Criteria.

C1. The candidate is a book.
C2. The candidate involves a group of powerful people who meet somewhere hidden to plan the simultaneous killing of important people across an entire continent.
C3. The candidate involves that group needing to eliminate one person who keeps getting in their way before proceeding.
C4. The candidate involves that group blackmailing a woman because she killed someone and hid the body somewhere disgusting.
C5. The candidate involves that woman being beaten and sent bloodied to the person that group wants dead as bait, exploiting his compassionate nature.
C6. The candidate involves that group wanting that person's death to look like a natural medical condition.

Model responses.

Model	Candidates	Best candidate	Criteria met
DeepSeek V4 Pro	5	*The Big Four* (Agatha Christie)	2/6
Claude Opus 5	6	*The Wandering Jew* (Eugène Sue)	1/6
Claude Sonnet 5	5	*The Matarese Circle* (Robert Ludlum)	1/6
GPT-5.6 Luna	2	*The Big Four* (Agatha Christie)	1/6
Gemini 3.7 Flash	3	*The Wheel of Time* (Robert Jordan)	0/6
Qwen 3.8 Max	0	–	–
GPT-5.6 Sol	–	*run failed*	–

Why it fails. This query has the lowest criterion scores of the five. No candidate satisfies more than two of the six criteria, and candidates drift towards Western conspiracy fiction such as Fu Manchu, *The Big Four*, and *The Wheel of Time*. Qwen 3.8 Max returned no candidates, and GPT-5.6 Sol's run exhausted the model's context window (Appendix D.4).

Case 4: A famous namesake

Seed. *Fix You* (Vita Chambers), a Canadian dance-pop single [source] **Turns.** 2

> *I'm trying to figure out a song I have stuck in my head, yet I don't remember the words. It was on the radio all the time when I worked at Tim Horton's in the early 2010s and it was definitely that era's dance-pop style. The singer is a solo woman and I'm pretty sure she was a one-hit wonder. The title of the song sounds like another more famous song.*

Criteria.

C1. The candidate is a song.
C2. The candidate was played on the radio in the early 2010s.
C3. The candidate is in a dance-pop style.
C4. The candidate is performed by a solo artist.
C5. The candidate is performed by a woman.
C6. The candidate has a title that sounds like the title of another song.

Model responses.

Model	Candidates	Top candidate	Criteria met
Claude Opus 5	12	*Alone Again* (Alyssa Reid)	6/6
GPT-5.6 Sol	8	*Shut Up and Dance* (Victoria Duffield)	6/6
DeepSeek V4 Pro	8	*Mr. Saxobeat* (Alexandra Stan)	5/6
Claude Sonnet 5	6	*Call Me Maybe* (Carly Rae Jepsen)	5/6
Gemini 3.7 Flash	6	*Shut Up and Dance* (Victoria Duffield)	6/6
Qwen 3.8 Max	4	*Call Me Maybe* (Carly Rae Jepsen)	5/6
GPT-5.6 Luna	1	*Alone Again* (Alyssa Reid)	6/6

Why it fails. The decisive clue is C6, and it helps a person but not a search engine: searching the title returns the far more famous song it resembles, which the query never names. Four of the seven models return a top candidate satisfying all six criteria, and none is the seed.

Case 5: Unanimous convergence on a famous referent

Seed. *Chłopaki nie płaczą* (2000), a Polish comedy [source] **Turns.** 3

> *A movie, there is this guy that isn't directly associated with the mafia at the start of the movie, but then interfers with their affairs. There is this famous quote in it, something about not trusting a woman and cutting hand. The film is pretty old.*

Criteria.

C1. The candidate is a film.
C2. The candidate has a male character who is not directly associated with the mafia at the start of the story.
C3. The candidate has that same male character interfering with the mafia's affairs.
C4. The candidate includes a quote about not trusting a woman and cutting hand off.

Model responses.

Model	Candidates	Top candidate	Criteria met
Claude Opus 5	10	*The Godfather* (1972)	3/4
GPT-5.6 Sol	7	*Casino* (1995)	3/4
Claude Sonnet 5	6	*The Godfather* (1972)	3/4
DeepSeek V4 Pro	6	*The Godfather* (1972)	3/4
Qwen 3.8 Max	6	*The Godfather* (1972)	3/4
Gemini 3.7 Flash	4	*The Godfather* (1972)	3/4
GPT-5.6 Luna	1	*The Yakuza* (1974)	3/4

Why it fails. Five of the seven models return *The Godfather*, and the other two return *Casino* and *The Yakuza*. Every top candidate satisfies three of the four criteria, failing only the quote, which is the one clue that would have separated them. The models agree on the wrong answer: the cues match a dominant English-language film so closely that none reaches the Polish one.

J SOURCE ANALYSIS

Because underspecified queries do not determine where models should search, we examine source selection as part of their search behaviour. We analyse both the sources models request and the pages they read.

Method. We measure two channels separately. A *read* is a URL a model opened with the `read` tool. We exclude failed fetches and infrastructure domains (caches, image proxies, and reader endpoints), and deduplicate by (model, query, domain), so a read share is the fraction of model-query pairs that opened a domain at least once. A *request* is a domain a model named with a `site:` operator in a search query, excluding negated operators; a request share is over operators. Exa does not honour `site:`, so requests never filtered results and record only where a model expected an answer to live. Each Open Discovery query carries a topic label from the generator's domain field, normalised into eleven topics.

Reading concentrates on Wikipedia. Wikipedia is opened in 42.6% of Open Discovery pairs and 32.3% of Entity Recall pairs, roughly nine to eleven times as often as the next domain, although models read from 3,401 and 2,197 distinct domains (Table 15). It is the most-read domain for every model, ranging from 17.6% of GPT-5.6 Sol's pairs to 53.3% of DeepSeek V4 Pro's (Figure 23).

Requests track the information need. In Open Discovery, models request the institution that holds the relevant kind of record: 56.2% of business requests name `sec.gov`, 28.7% of science requests name `ptacts.uspto.gov`, and heritage, health, and legislation requests name UNESCO, the FDA, and Congress (Table 16). These domains are topic-specific: 97 to 100% of all requests naming each come from a single topic, whereas Wikipedia, requested 738 times, draws no more than 34% of its requests from any one topic. This pattern comes from four models, GPT-5.6 Sol, GPT-5.6 Luna, Gemini 3.7 Flash, and DeepSeek V4 Pro, which issue almost all Open Discovery operators (Table 17).

Entity Recall requests target recall communities. Reddit is the most-requested domain in Entity Recall, taking 485 of 2,240 operators (21.7%) across 114 queries, and it appears among the five most-requested domains for all seven models (Figure 24). Of the 485 requests, 270 name a specific community, and 251 of those name one of four communities for identifying half-remembered media: `r/tipofmytongue` (147, including the abbreviation `tomt`), `r/whatsthatbook` (68), `r/tipofmyjoystick` (29), and `r/namethatsong` (7). A representative request:

```
site:reddit.com tipofmytongue film half white girl absent father lives
with her Mexican family
```

Request behaviour varies across models and tasks. Operator use varies by more than an order of magnitude across models and changes with the task (Table 17). Qwen 3.8 Max carries a `site:` operator in 3.3% of its Open Discovery pairs but 34.1% of its Entity Recall pairs, while GPT-5.6 Sol falls from 94.2% to 62.6%. Qwen 3.8 Max also ran at a different reasoning effort in each task (Table 8), so its change is not attributable to the task alone.

K DIVERSITY OF CANDIDATE SOLUTIONS ACROSS AND WITHIN MODELS

We are additionally interested in characterizing the diversity of the candidate solution space produced by each model, beyond whether models retrieve the correct entity. We therefore study diversity from two complementary perspectives: across models, to measure how much different models overlap in the candidates they propose for the same query, and within models, to measure how varied a single model's own candidate set is. For the across-model analysis, we quantify lexical overlap using Jaccard similarity over progressively larger candidate prefixes. For the within-model analysis, we measure both lexical diversity, defined as $1-$ mean pairwise Jaccard similarity, and semantic diversity, defined as $1-$ mean pairwise cosine similarity between candidate embeddings. We evaluate how within-model diversity changes as additional ranked candidates are considered, up to each model's median candidate-set size, and also examine the relationship between lexical and semantic diversity at that representative set size.

In Figure 25, we show the lexical overlap between candidate solutions produced by different models. Agreement is already modest when considering only the first word of each candidate and decreases

Open Discovery		Entity Recall	
Domain	**Share**	**Domain**	**Share**
Most read (share of model-query pairs opening the domain)			
en.wikipedia.org	42.6%	en.wikipedia.org	32.3%
sec.gov	4.8%	imdb.com	2.9%
pmc.ncbi.nlm.nih.gov	3.8%	kirkusreviews.com	2.9%
theguardian.com	3.5%	youtube.com	2.9%
bbc.com	2.7%	gutenberg.org	2.7%
Most requested (share of site: operators naming the domain)			
en.wikipedia.org	8.3%	reddit.com	21.7%
sec.gov	5.5%	youtube.com	3.9%
grammy.com	2.1%	goodreads.com	3.3%
olympedia.org	2.1%	en.wikipedia.org	3.1%
whc.unesco.org	2.0%	scifi.stackexchange.com	2.9%
Model-query pairs	1,536		1,186
Distinct domains read	3,401		2,197
site: operators	8,943		2,240

Table 15: **Wikipedia dominates reading in both tasks, while requests are task-specific.** Read shares are over model-query pairs; request shares are over site: operators, so the two blocks are not comparable.

Topic	Operators	Leading domain	Share of topic	Exclusivity
Business	878	sec.gov	56.2%	99.8%
Science	529	ptacts.uspto.gov	28.7%	100%
Geography	657	whc.unesco.org	26.6%	98.3%
Health	294	fda.gov	19.4%	100%
History	540	eap.bl.uk	16.1%	100%
Sports	1,293	medalbase.com	13.1%	100%
Politics	912	congress.gov	12.9%	100%
Film / TV	878	ilcinemaritrovato.it	12.1%	100%
Gaming	388	nichebarrier.com	11.1%	100%
Music	1,821	grammy.com	10.3%	100%
Literature	706	thebookerprizes.com	7.2%	100%

Table 16: **Models request the institution that holds each kind of record.** Open Discovery site: requests by query topic, excluding en.wikipedia.org. *Share of topic* is the fraction of that topic's operators naming the domain; *exclusivity* is the fraction of all operators naming the domain that come from this topic.

sharply as more of the candidate text is included, indicating that different models increasingly diverge in the solutions they propose. In Figure 26, we show how diversity evolves within each model as more ranked candidates are considered. Lexical diversity is consistently very high, while semantic diversity is lower, and both increase only modestly with additional candidates, suggesting that most of the candidate-space diversity is established among the first few solutions. In Figure 27, lexical diversity is concentrated near its maximum for most queries, while semantic diversity varies considerably within this high-lexical-diversity regime. This suggests that lexical diversity quickly saturates and cannot distinguish between candidate sets that differ substantially in semantic coverage.

Model	Open Discovery pairs with `site:` (%)	Entity Recall pairs with `site:` (%)
GPT-5.6 Sol	94.2	62.6
GPT-5.6 Luna	87.8	53.7
Gemini 3.7 Flash	74.6	73.4
DeepSeek V4 Pro	58.4	36.3
Claude Sonnet 5	9.0	8.5
Claude Opus 5	5.4	14.5
Qwen 3.8 Max	3.3	34.1

Table 17: **`site:` use varies widely across models and changes with the task.** Share of each model's query pairs containing at least one `site:` operator.

Parametric Baseline Prompt

```
You are an assistant answering from your own knowledge. You have no web search
or page-reading tools on this task.

The user is trying to recall something from memory using an abstract or
underspecified description. They know what they're looking for and will recognise
it among a set of candidates once they see it, even if they cannot specify all
the constraints and concrete details required to retrieve it.

Your task is to understand the query and retrieve candidates that satisfy what
the user is looking for.

For each candidate, confidence is your calibrated estimate (0 to 100) that it
is correct, not a ranking.

Here is the User Query: "{query}"

Return only valid JSON in this shape:
   {
      "candidates": [
         {
            "candidate":  "the thing or combination of things the query
describes",
            "confidence": 0-100,
            "key_evidence": "specific facts that support this candidate"
         }
      ]
   }
```

Figure 18: **Parametric Baseline Prompt.**

Verifier prompt (abridged)

SYSTEM INSTRUCTION

You are a rigorous web-based fact-checking verifier with web search and page-reading tools.

You are given a candidate entity proposed as an answer, a list of constraints it is claimed to satisfy, and the seed entity – the intended answer the constraints were written from, and which satisfies all of them by construction. Your task is to determine whether the candidate satisfies each constraint, verifying every one independently and rigorously against the web.

Candidate entity: "{candidate_name}" Seed entity: "{seed_entity}"
Constraints: {constraints_list}

1. Seed shortcut. First decide whether the candidate resolves to the same underlying entity as the seed – and the same version, edition, adaptation, installment and release. Exact string equality is not required. If it does, mark every constraint "YES" with the evidence text "Candidate matches the seed entity from which the constraints were generated.", set "entity_found" true, and return without searching. Do not take the shortcut for a different remake, edition, adaptation, sequel, release year or language version. If you are meaningfully unsure, skip the shortcut and verify normally.

2. Identify the candidate. Establish exactly which entity it refers to, accounting for alternate, translated and variant titles, aliases, editions, remakes, individuals versus groups, and franchise versus single installment. If no credible evidence shows the candidate exists, set "entity_found" false and mark every constraint "NO". Never invent an entity to make the constraints fit.

3. Verify each constraint independently. Treat each as a standalone factual proposition. Do not infer one constraint from another, credit a claim for being plausible, treat the constraint text itself as evidence, or rely on unsupported memory. The seed is context, not evidence: use it only to read an ambiguous constraint as intended and to tell similarly named entities apart, never to support a claim about the candidate. Every component must be supported. Read the wording literally: film versus television series, prince versus king, country of production versus setting, protagonist versus supporting character. Evidence must belong to the specific version at hand; do not carry facts across remakes, sequels or later editions.

4. Sources. Prefer official sources; then publishers, studios, broadcasters, archives and institutional sources; then reputable reference databases; then established publications. Use weaker sources only when nothing stronger exists, and say so.

5. Scoring. "YES" – credible evidence clearly supports the whole constraint. "NO" – evidence contradicts it, supports only part of it, is ambiguous, requires substantial inference, or could not be found; say which. Absence of evidence is not disproof, so never call an unverified claim false, but under this binary scheme an unverified or ambiguous constraint is still "NO". Be conservative: never credit vague similarity, thematic resemblance, facts from another version, or claims that hold only of the seed.

Output. Valid JSON only, one entry per constraint, in the order supplied, identified by its "C1", "C2", . . . label. Cite the URL relied on in "source_url" and never fabricate a source, URL or quote.

```
{"entity_found": true, "constraints": [{"constraint_id": "C1",
   "satisfied": "YES or NO", "evidence": "<concise factual justification>",
   "source_url": "https://example.com"}]}
```

Figure 19: **Verifier prompt.** Gemini 3.7 Flash judges each candidate constraint by constraint against the web. Abridged: the full prompt carries worked examples for the seed shortcut, the literal-reading rule and each category of NO.

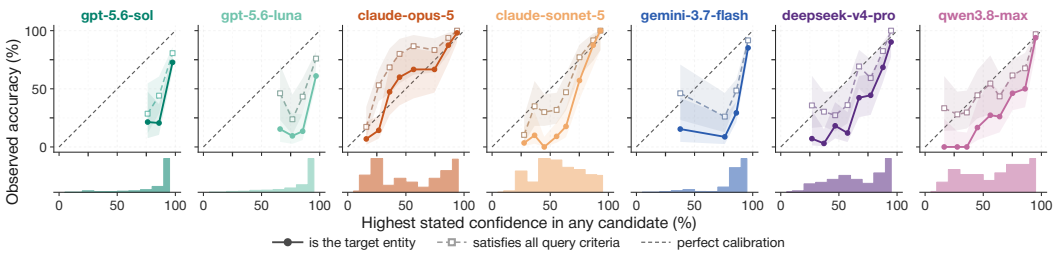


Figure 20: **Overconfidence decreases but persists when confidence is scored against criterion validity.** Reliability of the highest-confidence candidate under the identity and criterion outcomes in the search setting.

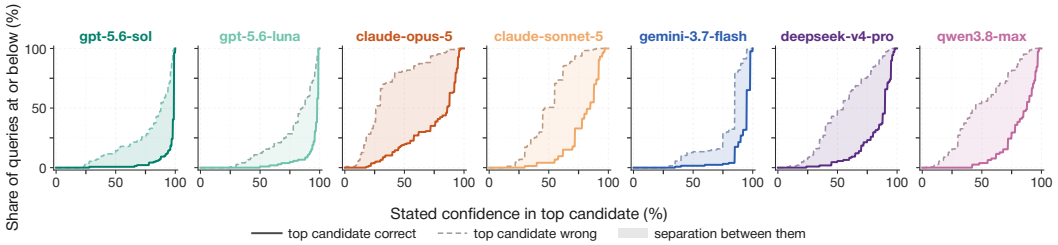


Figure 21: **Models discriminate correct from incorrect candidates despite differences in calibration.** Distribution of stated confidence for highest-confidence candidates that do and do not match the seed entity.

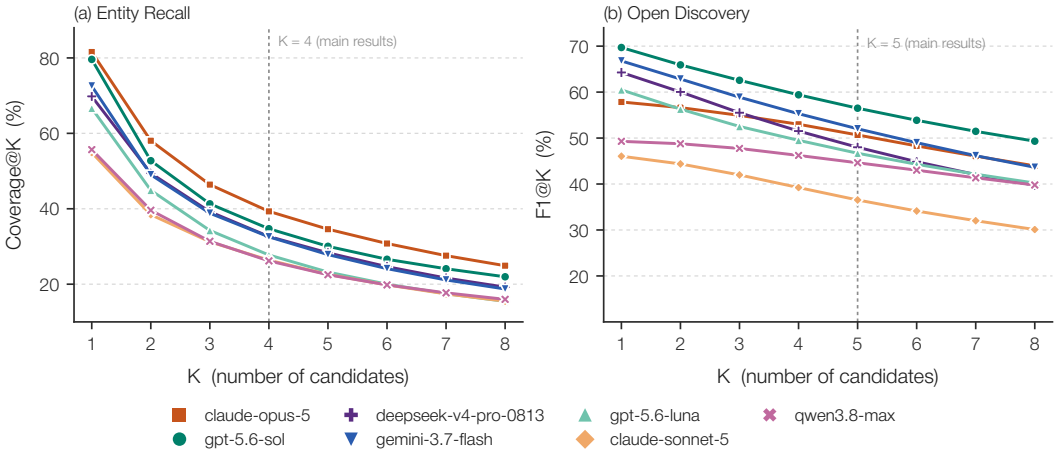


Figure 22: **Ranking is stable on Entity Recall and reorders with candidate count on Open Discovery.** (a) Coverage@K on Entity Recall and (b) F1@K on Open Discovery, for $K = 1$ to 8. Dashed lines mark the K used in the main results.

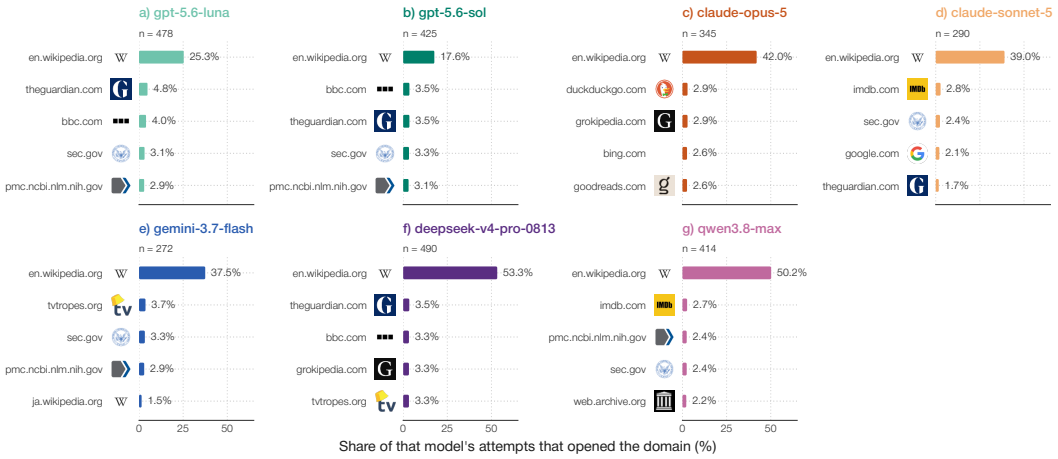


Figure 23: **Wikipedia is the most-read domain for every model.** Each model's five most-read domains, as a share of that model's query pairs, pooled over both tasks; n is the model's number of pairs.

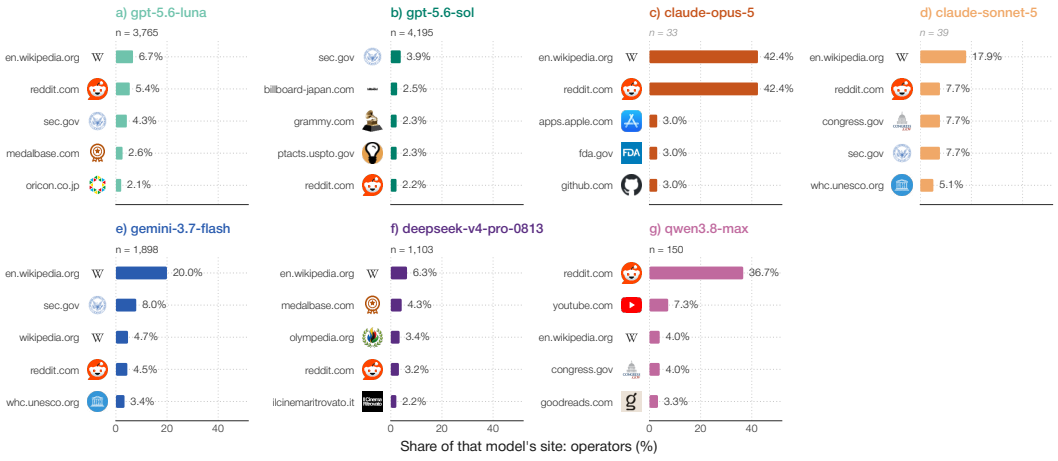


Figure 24: **Reddit is among the five most-requested domains for every model.** Each model's five most-requested domains, as a share of that model's `site:` operators, pooled over both tasks. A greyed n marks Claude Opus 5 (33 operators) and Claude Sonnet 5 (39), whose shares are unstable at that volume.

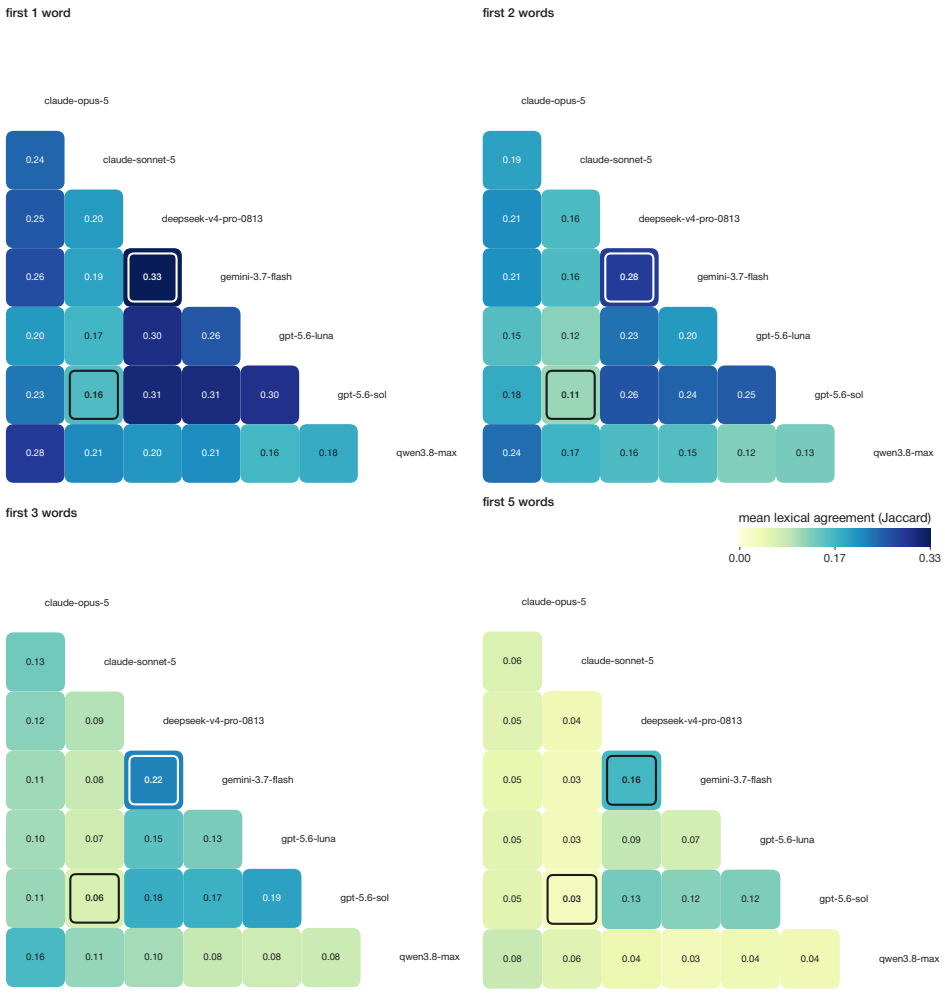


Figure 25: **Models show limited lexical agreement, which decreases as more of each candidate is considered.** Pairwise heatmaps report mean Jaccard similarity between candidate solutions produced by different models for the same queries, computed over the first 1, 2, 3, and 5 words of each candidate. Lower values indicate less lexical overlap between the candidate spaces explored by different models.

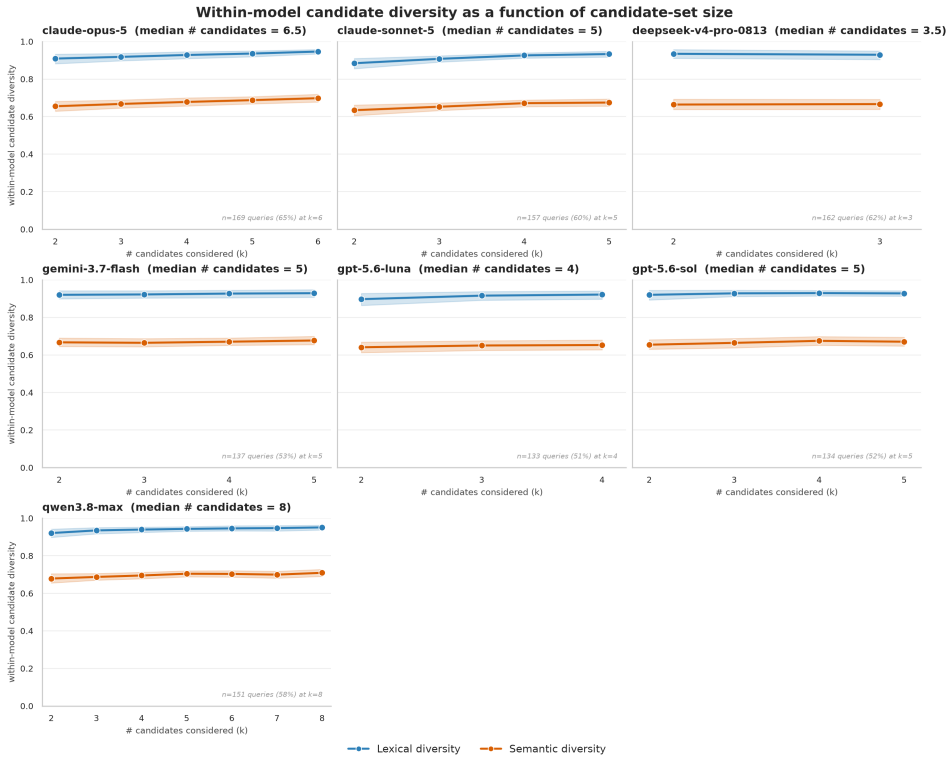


Figure 26: **Most within-model diversity is already present among the first few candidates.** For each model, we measure lexical diversity (1− mean pairwise Jaccard similarity) and semantic diversity (1− mean pairwise cosine similarity) among its top-k candidates. We vary k from 2 up to the model-specific median candidate-set size and average diversity across queries; shaded regions denote uncertainty across queries.

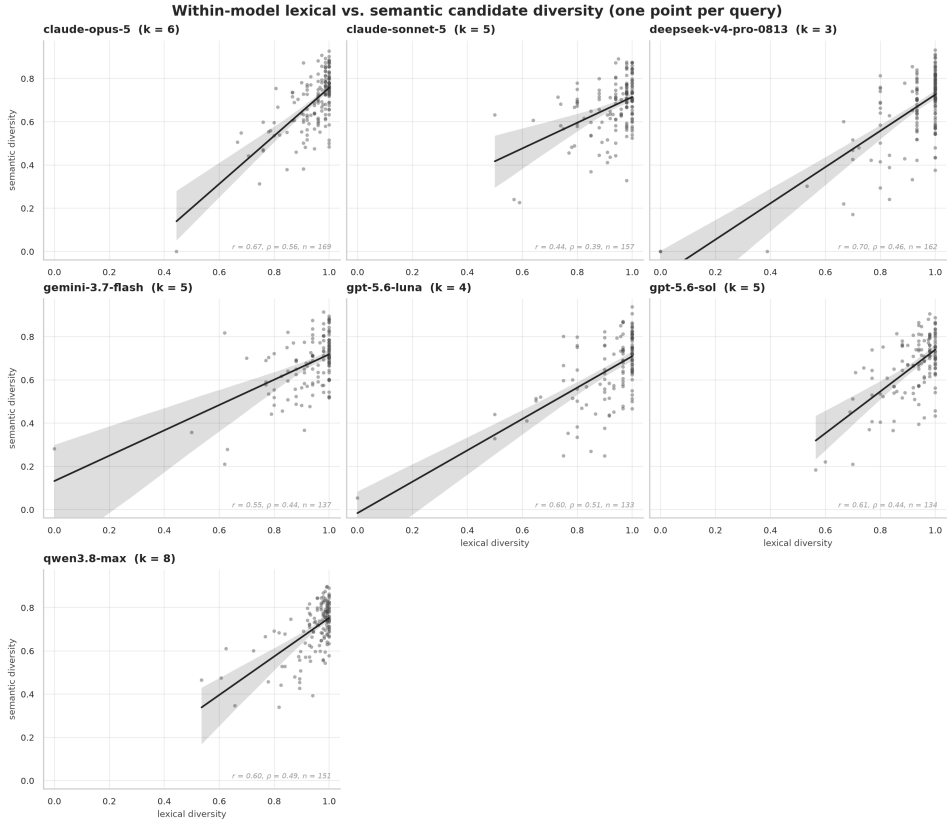


Figure 27: **High lexical diversity does not necessarily imply equally high semantic diversity.** Each point represents one query and measures lexical and semantic diversity among a model's top-k candidates, where k is the model-specific median candidate-set size. Lexical diversity is concentrated near its maximum for many queries, while semantic diversity remains substantially more variable, indicating that lexical diversity alone does not fully characterize semantic coverage.