OBLIQ-Bench: Exposing Overlooked Bottlenecks in Modern Retrievers with Latent and Implicit Queries

Diane Tchuindjo **Devavrat Shah** **Omar Khattab**

Massachusetts Institute of Technology, Cambridge, MA
{dianetc, devavrat, okhattab}@mit.edu

Abstract

Retrieval benchmarks are increasingly saturating, but we argue that efficient search is far from a solved problem. We identify a class of queries we call *oblique*, which seek documents that *instantiate a latent pattern*, like finding all tweets that express an implicit stance, chat logs that demonstrate a particular failure mode, or transcripts that match an abstract scenario. We study three mechanisms through which obliqueness may arise and introduce OBLIQ-Bench, a suite of five oblique search problems over real long-tail corpora. OBLIQ-Bench exposes an overlooked asymmetry between retrieval and verification, where reasoning LLMs reliably recognize latent relevance whenever relevant documents are surfaced, but even sophisticated retrieval pipelines fail to surface most relevant documents in the first place. We hope that OBLIQ-Bench will drive research into retrieval architectures that efficiently capture latent patterns and implicit signals in large corpora.[1]

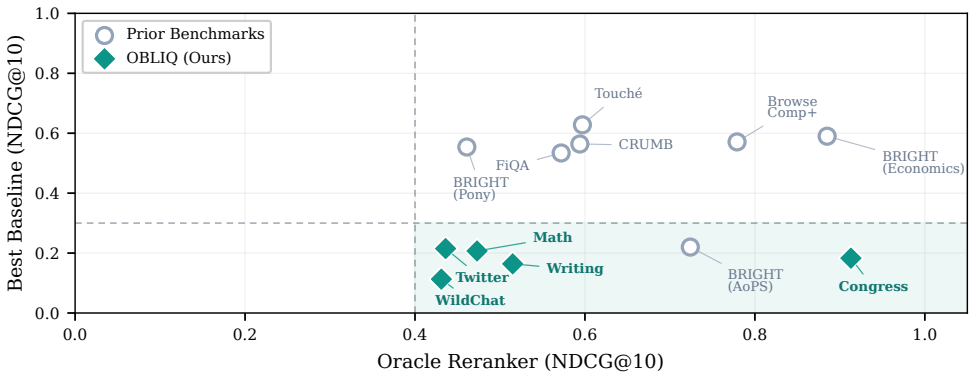


Figure 1: **Compared with prior benchmarks, relevant documents in OBLIQ-Bench are easy to recognize but much harder to retrieve.** Each point on this plot is a retrieval benchmark. The y axis shows the best NDCG@10 obtained by a suite of state-of-the-art retrieval systems and agentic multi-hop search pipelines. The x axis shows the NDCG@10 obtained when a reasoning model re-ranks a very large pool of hard distractors infused with the gold results. Prior benchmarks mostly sit near the diagonal: strong retrievers recover much of what the reasoning model can verify. In contrast, oblique tasks fall in the lower-right: the reasoning model can identify relevant documents, but even LLM-driven retrieval systems fail to surface them. More details in Sections 3 and 5.

[1] Data: https://huggingface.co/datasets/dianetc/OBLIQ-Bench

1 Introduction

Modern information retrieval (IR) benchmarks increasingly portray the existing IR paradigms as highly effective and rather indistinguishable. Across standard passage-ranking datasets like MS MARCO (Bajaj et al., 2016), BEIR (Thakur et al., 2021), MTEB (Muennighoff et al., 2023), and even BRIGHT (Su et al., 2025), signs of saturation are common and it is increasingly hard to observe more than a marginal difference in metrics like NDCG@k across strong dense retrievers (Karpukhin et al., 2020; Qu et al., 2021), late interaction models (Khattab and Zaharia, 2020), and LLM-based multi-hop agents (Yao et al., 2023; Chang et al., 2026).

In fact, in numerous existing IR benchmarks (Figure 1; top right), documents whose relevance can be verified by a reasoning model are, to a large extent, already easy to retrieve by efficient search algorithms. If scalable retrievers can nearly match the ceiling set by a prohibitive frontier LLM that can read the query and document together, what is left to solve?

We argue that this apparent saturation is an artifact of existing benchmarks, not a property of the problem of retrieval. There exist natural queries, ones that real users might pose and that ambitious downstream systems should support, on which current retrieval architectures score extremely poorly while a reasoning LLM scores substantially higher (Figure 1; bottom right). These queries share an informal property we call *obliqueness*: the attributes that determine relevance are latent and have little or no surface expression in the document.

Consider an analyst searching a corpus of tweets for any post p that *mock how users turn distant armed clashes into entertainment* without p expressing that explicitly. The relevant tweets may communicate their stance through irony or the choice of what to emphasize, but will not generally overlap directly with the query. Or consider an auditor searching Human–AI conversations for cases where a model *receives a formatting constraint, violates it several turns later, and never corrects itself.* The failure is evident in the transcript but might not be a topic that is acknowledged during the conversation. Or perhaps an ML expert who is trying to decontaminate training data may want to find *all problems that use the same proof strategy as a given question* despite different details.

Toward pushing the field to tackle such ambitious retrieval challenges, we make three contributions. First, we identify oblique retrieval as an overlooked regime for evaluating search systems. We

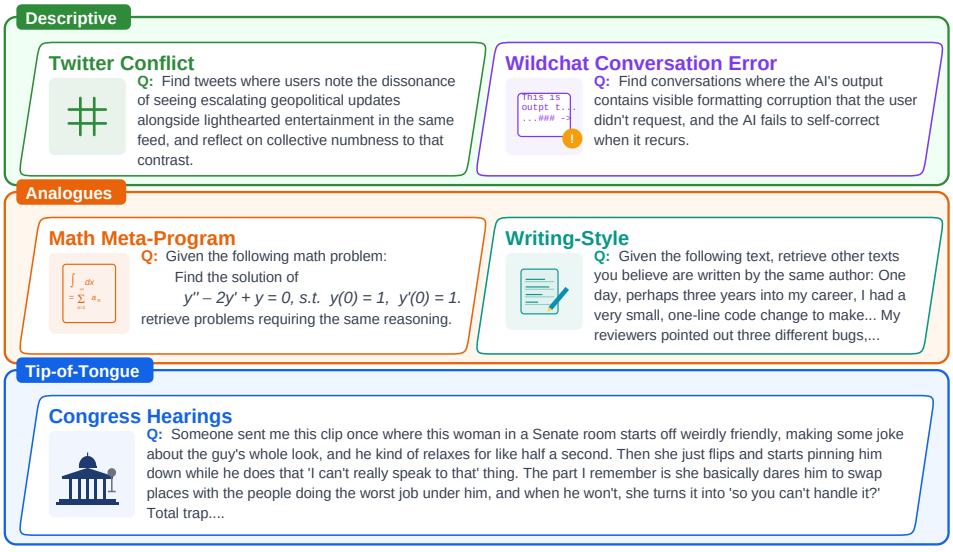


Figure 2: **Our five OBLIQ-Bench tasks span three types of oblique search queries.** Descriptive queries seek a latent property that can be inferred from document content, like tweets that subtly imply a detailed stance and Human–AI conversations that exhibit an implicit failure mode. Analogue queries seek all documents that share an archetype with the content of the query, despite differing in surface topic, like math problems that yield to the same proof technique or text snippets that share an author's stylistic fingerprint across topics. Lastly, tip-of-the-tongue queries match a fuzzy recollection to a specific obscure passage, like a transcript of a Congress hearing. In every task, relevance is verifiable by a reasoning model but hard for current retrieval systems.

formalize the gap between retrieval failure and verification success, or the *retrieval–verification asymmetry* (§3), as an empirical test for obliqueness, and use it to show that existing benchmarks often fail the test: they are not oblique with respect to existing methods (Figure 1).

Second, we develop OBLIQ-Bench (§4) by introducing five difficult retrieval tasks on top of five long-tail corpora (Figure 2). To obtain high-recall relevance judgments over a large corpus without exhaustively judging every query–document pair, we propose a pipeline for starting from a human-defined lens for reading a corpus—for example, implicit stance in tweets or failure modes in conversations—and allowing a reasoning LLM to (1) annotate the entire corpus over an affordable single pass, (2) cluster documents using these extracted latent attribute annotations, and (3) generate well-scoped queries that describe each cluster. Figure 2 shows representative example queries that we can produce, along with their relevance judgments, in this manner.

Finally, we evaluate lexical, dense, late interaction, and agentic retrievers across the five tasks (§5), and derive a set of five lessons about the existing IR landscape. All systems score poorly, and often close to zero NDCG@10, on every task. In contrast, a reasoning model applying tournament reranking over a very large pool of hard distractors (Figure 3) scales dramatically better with the pool size and reliably separates the relevant documents, confirming that the relevance signal exists but can be inaccessible to current approaches that do not apply joint reasoning over query and document. We release all corpora, queries, and relevance judgments, and hope OBLIQ-Bench drives work on new scalable retrieval approaches that can begin to attack such oblique queries.

2 Background and Related Work

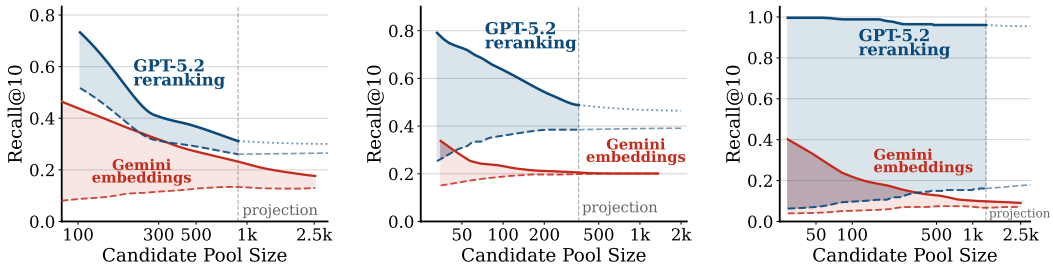
We are interested in scalable retrieval over a massive corpus $\mathcal{C} = \{d_1, \ldots, d_N\}$. At query time, a retriever receives a query q and an integer $k \ll N$, and must return an ordered list $d_{i_1}, \ldots, d_{i_k}$. The retriever may spend considerable work offline, say, to precompute representations or build an index, but its per-query cost should be sublinear in the corpus size, for instance $O(N^{0.5})$ total operations with at most $O(1)$ of them being calls to large LLMs, or at least it should have such small constants that querying a large $\mathcal{C}$ is possible. Modern retrieval paradigms fit this setting, not only lexical, dense retrievers, and late interaction methods but also highly sophisticated LLM search agents or query rewriting pipelines. We evaluate quality using standard IR metrics like NDCG@k and Recall@k.

OBLIQ-Bench focuses on queries where relevance is determined by a latent relation. *Oblique descriptive queries* seek documents that express a latent property implicitly, such as an implicit stance or behavioral failure mode. *Oblique analogue queries* seek documents that share an abstract structure with the query, such as a proof strategy or authorial style, despite differing in surface topic. Lastly, *oblique tip-of-the-tongue queries* seek an obscure passage from a partial, lossy, and abstract recollection. These mechanisms touch several lines of prior work, but in OBLIQ-Bench *the bottleneck appears very early in the search process*.

For instance, existing description-based retrieval queries (Ravfogel et al., 2024) search for sentences from abstract descriptions of their content, like matching "a change of career path" to "moved on to a manager career" which is a shallower linguistic lens. Existing tip-of-the-tongue and inferential queries typically seek a *popular entity* like a movie, book, or celebrity (Arguello et al., 2021; Lin et al., 2023; Mozafari et al., 2026), like matching "Spanish-speaking footballer holding the most titles" to passages about Lionel Messi, which is increasingly vulnerable to query re-writing via LLM parametric knowledge. Lastly, reasoning-intensive retrieval such as BRIGHT seek evidence documents that can be hard to verify but which are not systematically harder to retrieve into a top-k pool. OBLIQ-Bench seeks to make the latent relevance itself a first-stage search bottleneck and thus places the difficulty earlier (at scalably surfacing the documents in the first place), motivating the retrieval–verification diagnostic below. Appendix A contains an extended related work section.

3 Obliqueness and Retrieval–Verification Asymmetry

We seek to present more ambitious challenges to retrievers than current popular benchmarks do. To do so, we must guard against tasks that appear hard only because queries are ill-posed or where it is not possible to exceed a low score. In other words, we must find tasks t such that, for a set $\mathcal{R}$ of scalable retrieval systems of interest, the *retrieval–verification gap* is large. We define this gap as follows. For quality metric Q, let $gap(t) = V_t - R_t$ where $V_t = \max_{m \in \mathcal{M}} Q(m, t)$ represents the



(a) Twitter Conflict (b) Math Meta Program (c) Congress Hearings

Figure 3: **The retrieval–verification gap persists as hard candidate pools grow.** Recall@10 for the state-of-the-art dense retriever Gemini-2-Embedding and the GPT-5.2 reranker as as the size K of the candidate pool increases. For each model, dashed curves rank the retrieved pool as-is, giving a lower estimate that depends on the recall of the underlying pool. Solid curves inject missing gold documents into the pool before ranking, giving an upper estimate of how well the model recognizes positives once they are present. Faint and dotted GPT-5.2 segments after the vertical line are projected beyond the largest pool reranked. Across all three tasks, GPT-5.2 reranking remains far above Gemini-2-Embedding similarity in the upper and lower settings respectively.

best quality achieved over any model m in a powerful model class $\mathcal{M}$, which could ordinarily be prohibitively expensive, and where $R_t = \max_{r \in \mathcal{R}} Q(r, t)$ is the quality of the best-scoring $r \in \mathcal{R}$, an efficient retrieval system class.

Although it can be affordable (§4) to run a powerful model m on every document in $\mathcal{C}$ once or twice, we cannot exhaustively repeat this for every query q. Instead, we can ask some efficient system $r \in \mathcal{R}$ to fetch a set $K \gg k$ of documents and then rank these unordered K documents using a powerful model $m \in \mathcal{M}$.[2] Doing so establishes a *lower bound* on the quality of the best possible system $Q(\star, t)$. With modest assumptions about the stability of m, we can further approximate an *upper bound* on the quality of this same two-stage $\langle r, m \rangle$ system by ensuring that every missing "gold" document for a query q is injected with the (shuffled) K documents from r.

Using these two bounds, we can begin to estimate the retrieval–verification gap of a task t against a set $\mathcal{R}$ of state-of-the-art scalable retrieval systems. To make this estimation more reliable, we can monitor how the two bounds move as we scale K and observe how this movement for a powerful model $m \in \mathcal{M}$ differs from how retrievers $r \in \mathcal{R}$ move on the same quality metric Q.

Figure 3 demonstrates these patterns for a strong reasoning model m, which is set to GPT-5.2, and for the strongest frontier dense retriever in our experiments, Gemini-2-Embedding. The details of our listwise GPT-5.2 Tournament reranker and our other methods are discussed in §5. Here, we run a pool scaling experiment on three representative OBLIQ-Bench tasks: Twitter conflict (a descriptive task), Math Meta Program (an analogue task), and Congress Hearings (a tip-of-the-tongue task). For each task and each of the two methods, we evaluate Recall@10 as the candidate-pool budget increases, for two settings: standard and gold-infused.[3] Across all three tasks and in each of our two settings, GPT-5.2 reranking remains far above Gemini-2-Embedding as K grows.

Having validated this pooling approach, we apply it across many popular existing IR benchmarks and to our own five tasks in Figure 1. For each benchmark t, our estimate of the best scalable retrieval score R_t is on the x-axis and the verification score V_t is on the y-axis. The results suggest that many existing benchmarks are not oblique. On Touché-2020 (Bondarenko et al., 2020), FiQA-2018 (Maia et al., 2018), CRUMB (StackExchange QA; Killingback and Zamani 2025), and BrowseComp-Plus (Chen et al., 2025b), strong dense or agentic systems recover much of the score achieved by the gold-injected reranker. Only the AoPS slice of BRIGHT comes close to OBLIQ-Bench in our comparison, as Gemini-2-Embedding scores 0.20 NDCG@10 while GPT-5.2 reranking reaches 0.84

[2]This is inspired by pooling (Voorhees, 2007), TREC's standard approach for evaluating retrievers, in which "the judge reviews only the documents [in] the union of the set of X top-retrieved documents for that topic by each [retrieval system]".

[3]GPT-5.2 is evaluated over pools constructed from the union of BM25, Qwen3-Embedding-0.6B, and Qwen3-Embedding-4B, and Gemini-2-Embedding top-k' outputs. The Gemini embedding model is tested on equal pool sizes but without its own top-k because that would otherwise present it with its own hardest distractors and lower its score. Only the Congress task has a large distance between the upper and lower lines for GPT-5.2: since there's only a single relevant document per query in this task, the bottom line can never exceed the (low) success rate of the pool.

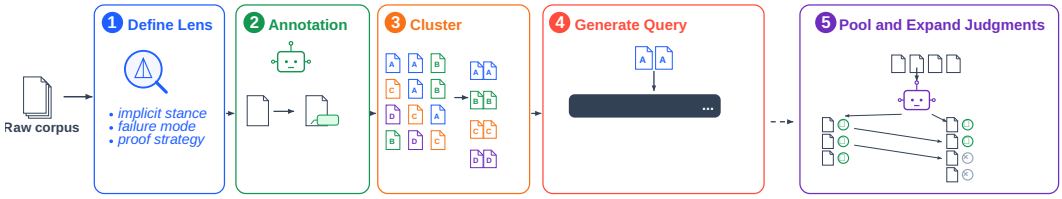


Figure 4: **Construction pipeline across OBLIQ-Bench.** A human defines a latent attribute (Stage 1). An LLM annotates documents through that lens (Stage 2), clusters attribute values (Stage 3), and generates abstract queries while forbidding source vocabulary (Stage 4). A pooling step optionally expands relevance judgments after evaluation (Stage 5). Writing-Style skips annotation and clustering because authorship is ground truth and Congress skips clustering, because each query in these two tasks targets one passage. All other tasks follow this entire pipline.

4 OBLIQ-Bench

We now introduce OBLIQ-Bench, a suite of five retrieval tasks designed to exhibit retrieval–verification asymmetry (§3) on real long-tail corpora. The tasks instantiate the three mechanisms of obliqueness from Figure 2. The central construction problem in OBLIQ-Bench is: *how should we obtain high-recall relevance judgments for latent properties at the scale of a large corpus, without judging every query–document pair?* If we simply asked a model to write queries for randomly sampled documents, we would risk producing artificially easy paraphrase tasks, or else having to judge the whole corpus after the fact to find the other documents that satisfy the same query.

Steps 1–2: Our construction pipeline (Figure 4) instead begins by *extracting the value of a latent attribute* from each document $d \in \mathcal{C}$ in a single upfront pass. Each of our five tasks defines an unobserved, and potentially sophisticated and costly to compute, document attribute $f(d)$ that determines relevance. In our five tasks, the function f is specified in natural language by a human: the stance implied by a tweet, the failure mode exhibited in a conversation, the proof strategy behind a math problem, the authorship fingerprint of a snippet, or the scenario that characterizes a transcript.

Step 3: Having produced $F = \{f(d) : d \in \mathcal{C}\}$, we then cluster the indexed values in F, producing document groups $\mathcal{G} = \{G_1, \ldots, G_m\}$, where each $G_j \subseteq \mathcal{C}$ contains documents with near-equivalent values of $f(d)$. We do this differently for each dataset, using multiple reasoning-LLM passes and, when useful, embedding models. The important point is that these models operate on the extracted attributes $f(d)$, not directly on the raw documents. This design is deliberate: it is far easier and more reliable to extract a scoped attribute from one document, and to merge similar attribute descriptions, than to search an arbitrarily large corpus with oblique queries. These passes merge near-synonymous labels in order to allow for latent queries to emerge naturally from the corpus.

Steps 4–5: For each cluster G_j, a reasoning LLM is tasked to verbalize the abstract shared attribute among all $d \in G_j$ into a textual query q while avoiding discriminative vocabulary and identifying details from the source documents. The initial positives for q_j are the documents in G_j, after any task-specific filtering and rejudging. After running a large set of retrievers, the top-k documents from all systems are pooled. For each query q, the reasoning LLM is tasked to read q, the known gold documents as anchors, and each of the unjudged candidates to identify any potential missed gold documents that can be added to the *pooled* setting of the respective dataset.

This recipe is adapted to the five tasks below, with expert human oversight throughout all steps to ensure quality and make task-level adjustments when necessary.In all tasks, we use real corpora, retain hard distractors, and remove metadata that would create search shortcuts. The Writing-Style task uses authorship as an external latent label, so no LLM annotation is needed, and the Congress Hearings task uses known target passages, so each query is a recollection of one passage rather than a cluster description. Dataset statistics appear in Table 1 and task-specific prompts and filtering details are presented in Appendix D.1–D.5. For all five tasks, example queries appear in both Figure 2 and example queries, positive documents, and negative documents appear in Table 5 in Appendix B.

Twitter-Conflict: Descriptive Retrieval over Implicit Stance. In this task, we ask systems to find *tweets that indicate a given stance on a topic and do so only implicitly*. We collect 72,122 English tweets via the X API full-archive search endpoint, scoped to a single geopolitical conflict beginning Feb 2026. GPT-5 classifies each tweet as `explicit_sentiment`, `news_repost`, or `implicit`.

Table 1: Dataset statistics. $\mathcal{Q}$ is the number of queries, $\mathcal{D}$ is corpus size, $\mathcal{D}_G^+/\mathcal{D}_P^+$ are average gold/pooled positives per query, and lengths are in characters.

Dataset Name	$\mathcal{Q}$	$\mathcal{D}$	Avg $\mathcal{D}_G^+$	Avg $\mathcal{D}_P^+$	Avg $\lvert\mathcal{Q}\rvert_{char}$	Avg $\lvert\mathcal{D}\rvert_{char}$
Twitter-Conflict	281	72,122	9.8	14.0	304	212
WildChat Errors	40	507,729	18.9	21.9	291	4,161
Math Meta Program	151	3,508	13.5	14.3	599	316
Writing-Style	512	10,389	9.00	–	3,057	2,315
Congress Hearings	254	213,650	1.00	–	808	2,545

Only the 7,918 implicit tweets can receive gold labels, while explicit and news tweets act as hard distractors. GPT-5 writes a translates each implicit tweet short into a description of a stance, which are then consolidated into canonical themes to produce queries, and then documents are rejudged to retain only score-2 query–tweet pairs (Appendix D.1). We also report pooled NDCG after asking the reasoning model to judge the top-10 retrieved results from the dense models and GPT 5.2 Tournament.

WildChat Errors: Descriptive Retrieval over LLM Interaction Transcripts. Here, we ask systems to retrieve Human–AI conversations that exhibit a named behavioral failure mode. We draw English conversations in WildChat-4.8M (Zhao et al., 2024), filter to 2025 conversations, serialize each as alternating user–assistant turns, and discard metadata such as model identity and topic annotations. GPT-5.4-nano performs a corpus-wide sweep and flags 14,743 conversations (2.9%) for errors. We cluster the raw labels in embedding space and use GPT-5 to collapse them into canonical failure types. GPT-5 then writes the queries, and a final rejudging pass retains only score-2 query–conversation pairs; we use the same pooled evaluation protocol as Twitter.

Math Meta-Program: Analogues via Shared Reasoning Technique. Here, the query is a problem whose solution uses a latent reasoning technique, and relevant documents are other problems that use the same technique across different mathematical topics. We draw problems and solutions from Putnam and other related undergraduate math competitions, the American Mathematical Monthly, and qualifying-exam sources. GPT-5 identifies the underlying meta-program of each solution, these labels are collapsed into canonical clusters, and one abstract query is generated for each.

Writing-Style: Analogues via Cross-Domain Authorship. We test whether systems can retrieve prose by the same author when topic is deliberately varied. To do so, we manually derive snippets from 64 researchers who post across unrelated subjects, scrub names and bylines, and split articles into snippets; authorship supplies the gold label. We add length-matched distractors from LWN.net, LessWrong, and Quanta Magazine to preserve register while removing easy source-contiguity cues. Each gold snippet serves as a query, and relevant documents are other snippets by the same author. Snippets from the same source post are masked at retrieval time, forcing systems to rely on cross-topic stylistic invariants rather than local context or topic. No pooled labels are used since relevance is determined by authorship.

Congress Hearings: Tip-of-the-Tongue over Transcript Scenarios. This task asks systems to recover a single obscure passage from a lossy recollection. We scrape GovInfo congressional hearing transcripts from the 110th through 119th Congresses, segment them into speaker-turn passages, and supplement them with ten high-profile technology hearings, including major antitrust, social media, and child-safety testimonies. Each query describes the target exchange's dynamic, emotional register, or rhetorical move while omitting names, dates, committees, and verbatim transcript phrasing. A hardening pass removes remaining identifiers. Each query has one gold passage.

5 Evaluation & Analyses

Having produced OBLIQ-Bench and validated that its query–document relevance relations are verifiable to a powerful reasoning model that operates as a ranker, we now ask what oblique retrieval tasks can teach us about the existing methods for scalable search.

We evaluate seven systems spanning five architecture families. For efficient single-stage retrieval, we evaluate lexical, dense, and late interaction approaches. We test BM25 (Robertson and Zaragoza, 2009) as the lexical baseline with default parameters in the `rank-bm225` Python library (Brown, 2020).For dense retrieval, we use three popular state-of-the-art embedding models that index the

Table 2: Results on the descriptive style queries. Namely, the Twitter-Conflict and WildChat tasks. Each metric shows Gold (G) / Pooled (P) evaluation. Key metrics highlighted.

		NDCG@10		NDCG@50		Recall		
Dataset	**Pipeline**	**G**	**P**	G	P	**@10**	@50	@100
Twitter	BM25	.000	.002	.001	.005	.000/.002	.002/.008	.002/.011
	LateOn 0.1B	.004	.010	.007	.016	.004/.008	.012/.025	.028/.047
	Qwen3-Embed-0.6B	.008	.012	.017	.033	.009/.009	.034/.062	.056/.093
	Qwen3-Embed-4B	.032	.039	.054	.088	.037/.037	.099/.148	.137/.199
	Gemini-2-Embedding	.068	.132	.101	.247	.071/.100	.169/.389	.231/.452
	GPT-5.2 Query Rewriter	.066	.139	.099	.201	.072/.116	.166/.282	.216/.369
	GPT-5.2 Multi-Hop Agent	.141	.215	.164	.255	.141/.176	.211/.302	.226/.317
	Oracle GPT-5.2 Tournament	.331	.436	.424	.555	.309/.342	.580/.674	.745/.812
WildChat	BM25	.004	.006	.004	.006	.003/.005	.003/.005	.003/.005
	LateOn 0.1B	.002	.003	.008	.007	.003/.005	.023/.014	.025/.015
	Qwen3-Embed-0.6B	.059	.073	.062	.064	.049/.046	.062/.059	.063/.061
	Qwen3-Embed-4B	.031	.059	.037	.062	.032/.049	.056/.077	.079/.095
	Gemini-2-Embedding	.057	.097	.078	.148	.059/.075	.123/.217	.151/.256
	GPT-5.2 Query Rewriter	.065	.095	.073	.111	.044/.048	.099/.142	.121/.171
	GPT-5.2 Multi-Hop Agent	.070	.113	.091	.098	.054/.077	.096/.128	.106/.137
	Oracle GPT-5.2 Tournament	.397	.431	.524	.557	.304/.308	.674/.680	.824/.837

corpus offline and retrieve by cosine similarity: **Gemini Embedding 2**, **Qwen3-Embedding-0.6B**, and **Qwen3-Embedding-4B** (Lee et al., 2025; Zhang et al., 2025). We also include results for a recent multi-vector late interaction model, **LateOn** (Sourty et al., 2026), with 149M parameters using PyLate (Chaffin and Sourty, 2025) for indexing with the PLAID (Santhanam et al., 2022) algorithm.

For agentic retrieval, we evaluate two types of pipelines, both of which pair GPT-5.2 reasoning with Gemini-2-Embedding dense retrieval. First, we evaluate a single-hop **GPT-5.2 Query Rewriter**, which tasks GPT-5.2 to reformulate the query once into a retrieval-optimized form, after which Gemini-2-Embedding retrieves the top 1,000 documents using the rewritten query. Second, we evaluate a **GPT-5.2 Multi-Hop Agent**, which iteratively applies query decomposition and document reading. In particular, in each hop for a total of four hops, GPT-5.2 generates a search query conditioned on the original query and any notes accumulated from prior hops, Gemini-2-Embedding retrieves the top-25 documents per hop, and GPT-5.2 reads each batch, selects pertinent candidates, and extracts observations to inform the next round. After four hops, GPT-selected candidates are promoted to top positions and the remaining retrieved-but-unselected documents fill the tail.

As discussed in §3, we establish an approximate bound on the attainable quality for each task using a powerful reasoning model, namely GPT-5.2, as a *listwise* ranker (Ma et al., 2023b; Sun et al., 2023; Chen et al., 2025a). Here, we construct a candidate pool by taking the union of approximately 300 top-k results from the union of Gemini-2-Embedding and both Qwen3-Embed models along with an injection of gold documents, per query. Refer to Figure 3 for a detailed discussion of this process. We refer to this system as **Oracle GPT-5.2 Tournament**. The procedure is a tournament-style listwise reranking algorithm, designed to expose GPT-5.2 only to small shuffled batches while focusing more compute on producing an approximate global head ranking. Appendix C presents the details of the algorithm we use for re-ranking arbitrarily large sets of documents.

5.1 Results by Mechanism

Descriptive queries. Table 2 shows a large retrieval–verification gap on both descriptive tasks. On Twitter-Conflict, Gemini-2-Embedding is the strongest single-stage retriever, reaching .132 pooled NDCG@10. The GPT-5.2 Query Rewriter is only marginally better, but the multi-hop agent improves more substantially, reaching .215 pooled NDCG@10. The oracle GPT-5.2 Tournament reaches .436 pooled NDCG@10. WildChat is similar in the size of the gap: the best non-oracle systems remain between .095 and .113 pooled NDCG@10, while the oracle tournament reaches .431. However, multi-hop search helps WildChat much less, plausibly because a tweet-level stance can sometimes be approached through a few alternative reformulations, whereas a conversation failure is an even more latent behavior that is distributed across many turns, and hence is hard to actively search for.

Table 3: Results on the analogue queries. Namely, the Math Meta-Program and Writing-Style tasks. Math shows Gold (G) / Pooled (P); Writing uses a single gold annotation. Compared to the other tables of results, here Recall@100 covers a disproportionately large fraction of each corpus, since the two corpora have only 3.5k and 10k documents respectively.

		NDCG@10		**NDCG@50**		**Recall**		
Dataset	**Pipeline**	**G**	**P**	G	P	**@10**	@50	@100
Math	BM25	.022	.029	.034	.025	.020/.029	.060/.029	.088/.029
	LateOn 0.1B	.112	.128	.141	.163	.088/.097	.213/.238	.285/.310
	Qwen3-Embed-0.6B	.116	.143	.149	.176	.070/.088	.219/.247	.309/.336
	Qwen3-Embed-4B	.095	.129	.119	.152	.078/.099	.172/.199	.253/.287
	Gemini-2-Embedding	.144	.147	.192	.217	.121/.156	.258/.296	.364/.398
	GPT-5.2 Query Rewriter	.142	.185	.198	.239	.138/.160	.324/.355	.414/.444
	GPT-5.2 Multi-Hop Agent	.161	.207	.210	.255	.145/.167	.307/.337	.387/.416
	Oracle GPT-5.2 Tournament	.279	.329	.399	.444	.276/.300	.610/.623	.790/.797
	Oracle GPT-5.2 Tournament+Soln	.434	.473	.546	.579	.406/.417	.752/.763	.885/.889
Writing	BM25	.077		.114		.062	.154	.208
	LateOn 0.1B	.105		.149		.087	.197	.263
	Qwen3-Embed-0.6B	.046		.060		.040	.075	.103
	Qwen3-Embed-4B	.033		.046		.026	.058	.080
	Gemini-2-Embedding	.164		.220		.132	.275	.357
	GPT-5.2 Query Rewriter	.018		.018		.008	.013	.017
	GPT-5.2 Multi-Hop Agent	.061		.059		.034	.037	.038
	Oracle GPT-5.2 Tournament	.515		.603		.449	.686	.797

Analogue queries. Table 3 shows that our analogue retrieval tasks are hard for different reasons. On Math Meta-Program, LateOn, Qwen3-Embed-0.6B, Gemini-2-Embedding, and the two GPT-5.2 retrieval pipelines all recover some signal, with the multi-hop agent reaching .207 pooled NDCG@10. The standard GPT-5.2 Tournament reaches .329, and when the tournament is also given solutions, however, it rises to .473 pooled NDCG@10. This confirms that the relevance relation depends on the (typically latent) proof structure. In contrast, on Writing-Style, BM25 beats both Qwen embedding models, LateOn beats BM25, and Gemini is the best single-stage retriever at .164 NDCG@10. This time, GPT-5.2 rewriting and multi-hop pipelines hurt substantially, perhaps because reformulating a prose snippet into retrieval keywords tends to preserve topic while destroying the stylistic signal that is the substance of retrieval for Writing-Style. The tournament, which can read query and candidate side by side, reaches .515 NDCG@10.

Tip-of-the-tongue queries. Table 4 gives the starkest version of the retrieval–verification asymmetry. Congress Hearings has exactly one gold passage per query, so Recall@k directly measures whether the target passage has been surfaced at all. BM25 never finds it at rank 10, Gemini-2-Embedding reaches only .079 Recall@10, and here the extremely small LateOn at 149M parameters outperforms every other single-stage retriever and reaches .102. This makes sense: being able to match local alignment between a transcript and the abstract scenario in the query could plausibly benefit from multi-vector late interaction. The GPT-5.2 Query Rewriter is essentially tied with LateOn, while the multi-hop agent improves to .185 Recall@10 and then plateaus: its Recall@10, Recall@50, and Recall@100 are all .185. In other words, the agent likely retrieves the target passage during one of its hops and places it near the top, and hence looking deeper in its final ranking does not uncover many additional successes. By contrast, the tournament reaches .913 NDCG@10 and perfect Recall@100.

5.2 Lessons

There is substantial headroom, but even the verifier is not perfect. Across all five tasks, the best non-oracle retrieval pipeline still score very poorly on NDCG@10 and trail considerably behind GPT-5.2 Tournament, though the gap is smaller on the Math task. The oracle reranker generally does well, though it is not perfect. We hope that this motivates and enables new retrieval architectures that can break through the bottleneck of these types of oblique queries and also work that seeks to improve the quality/cost tradeoffs of the oracle ranker (Appendix C) that we use.

Dense retrievers vary widely, but all of them fall short. Google's Gemini-2-Embedding is the strongest dense retriever on every task, and in several cases by a large margin. At the same time, the Qwen results are not monotonic in model size: Qwen3-Embed-0.6B beats Qwen3-Embed-4B

Table 4: Tip-of-tongue query results on Congress Hearings. Each query has exactly one gold passage; no pooled annotations are needed. Key metrics highlighted.

	NDCG		Recall		
Pipeline	**@10**	@50	**@10**	@50	@100
Congress Hearings					
BM25	.000	.002	.000	.008	.016
LateOn 0.1B	.083	.094	.102	.149	.185
Qwen3-Embed-0.6B	.006	.014	.012	.047	.055
Qwen3-Embed-4B	.040	.047	.063	.096	.122
Gemini-2-Embedding	.059	.066	.079	.114	.126
GPT-5.2 Query Rewriter	.084	.092	.102	.134	.158
GPT-5.2 Multi-Hop Agent	.183	.183	.185	.185	.185
Oracle GPT-5.2 Tournament	.913	.919	.957	.988	1.00

on WildChat, Math, and Writing-Style, perhaps due to the subtle ranking criteria in oblique queries. Given the success of the proprietary Gemini-2-Embedding model, we might infer that this is a signal for the value of the training data for such non-conventional retrieval problems.

Lexical and late-interaction systems fail in different ways. BM25 is mostly not competitive, which is unsurprising given that OBLIQ-Bench was built to remove direct lexical shortcuts. The one partial exception is Writing-Style, where BM25 beats both Qwen embedding models, perhaps because authorial style leaves some subtle lexical residue even across topics. LateOn shows the opposite profile: it is quite weak on the descriptive tasks, but punches well above its size on Math, Writing-Style, and Congress Hearings. This is especially clear on Congress, where a 149M-parameter late-interaction model is the best single-stage retriever. On the descriptive Twitter-Conflict task, LateOn often retrieves semantically natural topical matches that state the requested stance *explicitly*, but those are distractors because the model fails to interpret the instruction seeking only implicit matches. Token-level matching seems useful when the query and document share local scenario structure, but is not necessarily sufficient to make up for the lack of suitable training data when the relevant property is a diffuse stance or behavioral failure, like when relevance depends on nuanced instruction following at retrieval time.

Agentic search helps only when obliqueness can be translated into heuristic search actions. The multi-hop agent helps on Twitter and Congress, helps modestly on Math, barely helps on WildChat, and hurts considerably on Writing-Style. Iterative reformulation can help when the latent target can be approached through several alternative phrasings, as in implicit tweet stances or lossy recollections of a congressional exchange. It does not appear to help when the signal is distributed across a long conversation, and it can actively damage retrieval when the signal is orthogonal to topic, as in Writing-Style. Query rewriting might hence not be a silver-bullet solution to obliqueness.

6 Conclusion

OBLIQ-Bench is built around a simple idea: as reasoning models become increasingly more powerful by leveraging highly expressive architectures and scaling inference-time compute up for processing each of their prompts, we can begin to define very hard retrieval problems where reasoning LLMs have no trouble recognizing even very subtle cases of relevant documents when they are shown, but current scalable retrieval systems cannot surface them from the corpus in the first place. OBLIQ-Bench seeks to put the search bottleneck into first-stage retrieval. This can, as far as our results suggest, be resistant to existing approaches for query rewriting, agentic multi-hop search, and bigger and better embedding models. Across five long-tail tasks, we find this pattern for implicit stances, behavioral failures, proof strategies, writing styles, and lossy recollections. Our construction pipeline turns these latent properties into scalable relevance judgments and the results suggest that the next frontier for retrieval might need to be architectures that make latent document attributes available at search time. Lastly, we acknowledge the use of assistance from generative AI tools, with all outputs reviewed and edited by the authors, in the preparation of portions of this paper.

Acknowledgments

This work used Expanse GPU at the San Diego Supercomputer Center (SDSC) through allocation CIS250733 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS; Boerner et al. 2023) program, which is supported by U.S. National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296. This work was supported in part by MIT OCP project 6954031 and `cmpnd.ai`, and used OpenAI API credits gifted by Laude Institute and Mixedbread. We thank Antoine Chaffin and Benjamin Clavié for offering feedback on a late version of this manuscript.

References

Jaime Arguello, Adam Ferguson, Emery Fine, Bhaskar Mitra, Hamed Zamani, and Fernando Diaz. 2021. 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* (Canberra ACT, Australia) *(CHIIR '21)*. Association for Computing Machinery, New York, NY, USA, 5–14. doi:10.1145/3406522.3446021

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. MS MARCO: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268* (2016).

Timothy J. Boerner, Stephen Deems, Thomas R. Furlani, Shelley L. Knuth, and John Towns. 2023. ACCESS: Advancing Innovation: NSF's Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support. In *Practice and Experience in Advanced Research Computing (PEARC '23)*. Association for Computing Machinery, New York, NY, USA. doi:10.1145/3569951.3597559

Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, et al. 2020. Overview of touché 2020: argument retrieval. In *International Conference of the Cross-Language Evaluation Forum for European Languages*. Springer, 384–395.

Dorian Brown. 2020. *Rank-BM25: A Collection of BM25 Algorithms in Python*. doi:10.5281/zenodo.4520057

Antoine Chaffin and Raphaël Sourty. 2025. PyLate: Flexible Training and Retrieval for Late Interaction Models. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management, CIKM 2025, Seoul, Republic of Korea, November 10-14, 2025*, Meeyoung Cha, Chanyoung Park, Noseong Park, Carl Yang, Senjuti Basu Roy, Jessie Li, Jaap Kamps, Kijung Shin, Bryan Hooi, and Lifang He (Eds.). ACM, 6334–6339. doi:10.1145/3746252.3761608

Jonathan D. Chang, Andrew Drozdov, Shubham Toshniwal, Owen Oertell, Alexander Trott, Jacob Portes, Abhay Gupta, Pallavi Koppol, Ashutosh Baheti, Sean Kulinski, Ivan Zhou, Irene Dea, Krista Opsahl-Ong, Simon Favreau-Lessard, Sean Owen, Jose Javier Gonzalez Ortiz, Arnav Singhvi, Xabi Andrade, Cindy Wang, Kartik Sreenivasan, Sam Havens, Jialu Liu, Peyton DeNiro, Wen Sun, Michael Bendersky, and Jonathan Frankle. 2026. KARL: Knowledge Agents via Reinforcement Learning. arXiv:2603.05218 [cs.AI] https://arxiv.org/abs/2603.05218

Yiqun Chen, Qi Liu, Yi Zhang, Weiwei Sun, Xinyu Ma, Wei Yang, Daiting Shi, Jiaxin Mao, and Dawei Yin. 2025a. Tourrank: Utilizing large language models for documents ranking with a tournament-inspired strategy. In *Proceedings of the ACM on Web Conference 2025*. 1638–1652.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Sahel Sharifymoghaddam, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Nandan Thakur, Crystina Zhang, Luyu Gao, Wenhu Chen, and Jimmy Lin. 2025b. BrowseComp-Plus: A More Fair and Transparent Evaluation Benchmark of Deep-Research Agent. arXiv:2508.06600 [cs.CL] https://arxiv.org/abs/2508.06600

Yung-Sung Chuang, Wei Fang, Shang-Wen Li, Wen-tau Yih, and James Glass. 2023. Expand, Rerank, and Retrieve: Query Reranking for Open-Domain Question Answering. In *Findings of the Association for Computational Linguistics: ACL 2023*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 12131–12147. doi:10.18653/v1/2023.findings-acl.768

Yair Feldman and Ran El-Yaniv. 2019. Multi-Hop Paragraph Retrieval for Open-Domain Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Anna Korhonen, David Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, Florence, Italy, 2296–2309. doi:10.18653/v1/P19-1222

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. arXiv:2011.01060 [cs.CL] https://arxiv.org/abs/2011.01060

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. arXiv:2112.09118 [cs.IR] https://arxiv.org/abs/2112.09118

Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. 2020. HoVer: A Dataset for Many-Hop Fact Extraction And Claim Verification. In *Findings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 6769–6781. doi:10.18653/v1/2020.emnlp-main.550

Omar Khattab and Matei Zaharia. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 39–48. doi:10.1145/3397271.3401075

Julian Killingback and Hamed Zamani. 2025. Benchmarking Information Retrieval Models on Complex Retrieval Tasks. *arXiv preprint arXiv:2509.07253* (2025).

Jinhyuk Lee, Feiyang Chen, Sahil Dua, Daniel Cer, Madhuri Shanbhogue, Iftekhar Naim, Gustavo Hernández Ábrego, Zhe Li, Kaifeng Chen, Henrique Schechter Vera, Xiaoqi Ren, Shanfeng Zhang, Daniel Salz, Michael Boratko, Jay Han, Blair Chen, Shuo Huang, Vikram Rao, Paul Suganthan, Feng Han, Andreas Doumanoglou, Nithi Gupta, Fedor Moiseev, Cathy Yip, Aashi Jain, Simon Baumgartner, Shahrokh Shahi, Frank Palma Gomez, Sandeep Mariserla, Min Choi, Parashar Shah, Sonam Goenka, Ke Chen, Ye Xia, Koert Chen, Sai Meher Karthik Duddu, Yichang Chen, Trevor Walker, Wenlei Zhou, Rakesh Ghiya, Zach Gleicher, Karan Gill, Zhe Dong, Mojtaba Seyedhosseini, Yunhsuan Sung, Raphael Hoffmann, and Tom Duerig. 2025. Gemini Embedding: Generalizable Embeddings from Gemini. arXiv:2503.07891 [cs.CL] https://arxiv.org/abs/2503.07891

Kevin Lin, Kyle Lo, Joseph Gonzalez, and Dan Klein. 2023. Decomposing complex queries for tip-of-the-tongue retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. 5521–5533.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023a. Query Rewriting in Retrieval-Augmented Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 5303–5315. doi:10.18653/v1/2023.emnlp-main.322

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

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: financial opinion mining and question answering. In *Companion proceedings of the the web conference 2018*. 1941–1942.

Jamshid Mozafari, Hamed Zamani, Guido Zuccon, and Adam Jatowt. 2026. Inferential Question Answering. In *Proceedings of the ACM Web Conference 2026*. 2384–2395.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive Text Embedding Benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, Andreas Vlachos and Isabelle Augenstein (Eds.). Association for Computational Linguistics, Dubrovnik, Croatia, 2014–2037. doi:10.18653/v1/2023.eacl-main.148

Yingqi Qu, Yuchen Ding, Jing Liu, Kai Liu, Ruiyang Ren, Wayne Xin Zhao, Daxiang Dong, Hua Wu, and Haifeng Wang. 2021. RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2021 Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (Eds.). Association for Computational Linguistics, Online, 5835–5847. doi:10.18653/v1/2021.naacl-main.466

Shauli Ravfogel, Valentina Pyatkin, Amir DN Cohen, Avshalom Manevich, and Yoav Goldberg. 2024. Description-Based Text Similarity. arXiv:2305.12517 [cs.CL] https://arxiv.org/abs/2305.12517

Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (April 2009), 333–389. doi:10.1561/1500000019

Keshav Santhanam, Omar Khattab, Christopher Potts, and Matei Zaharia. 2022. PLAID: an efficient engine for late interaction retrieval. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 1747–1756.

Zhengliang Shi, Shuo Zhang, Weiwei Sun, Shen Gao, Pengjie Ren, Zhumin Chen, and Zhaochun Ren. 2024. Generate-then-Ground in Retrieval-Augmented Generation for Multi-hop Question Answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Lun-Wei Ku, Andre Martins, and Vivek Srikumar (Eds.). Association for Computational Linguistics, Bangkok, Thailand, 7339–7353. doi:10.18653/v1/2024.acl-long.397

Raphael Sourty, Antoine Chaffin, Orion Weller, Paulo Demoura, and Amelie Chatelain. 2026. DenseOn with the LateOn: Open State-of-the-Art Single and Multi-Vector Models. https://huggingface.co/blog/lightonai/denseon-lateon.

Hongjin Su, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han yu Wang, Liu Haisu, Quan Shi, Zachary S Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan O Arik, Danqi Chen, and Tao Yu. 2025. BRIGHT: A Realistic and Challenging Benchmark for Reasoning-Intensive Retrieval. In *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=ykuc5q381b

Duolin Sun, Meixiu Long, Dan Yang, Junjie Wang, Yecheng Luo, Yue Shen, Jian Wang, Hualei Zhou, Chunxiao Guo, Peng Wei, Jiahai Wang, and Jinjie Gu. 2026. DIVER: A Multi-Stage Approach for Reasoning-intensive Information Retrieval. arXiv:2508.07995 [cs.IR] https://arxiv.org/abs/2508.07995

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agents. In *The 2023 Conference on Empirical Methods in Natural Language Processing*. https://openreview.net/forum?id=3Q6LON8y2I

Nandan Thakur, Jimmy Lin, Sam Havens, Michael Carbin, Omar Khattab, and Andrew Drozdov. 2025. FreshStack: Building Realistic Benchmarks for Evaluating Retrieval on Technical Documents. arXiv:2504.13128 [cs.IR] https://arxiv.org/abs/2504.13128

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models. arXiv:2104.08663 [cs.IR] https://arxiv.org/abs/2104.08663

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multihop Questions via Single-hop Question Composition. arXiv:2108.00573 [cs.CL] https://arxiv.org/abs/2108.00573

Ellen M Voorhees. 2007. TREC: Continuing information retrieval's tradition of experimentation. *Commun. ACM* 50, 11 (2007), 51–54.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. BrowseComp: A Simple Yet Challenging Benchmark for Browsing Agents. arXiv:2504.12516 [cs.CL] https://arxiv.org/abs/2504.12516

Tomer Wolfson, Harsh Trivedi, Mor Geva, Yoav Goldberg, Dan Roth, Tushar Khot, Ashish Sabharwal, and Reut Tsarfaty. 2025. MoNaCo: More Natural and Complex Questions for Reasoning Across Dozens of Documents. arXiv:2508.11133 [cs.CL] https://arxiv.org/abs/2508.11133

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval. arXiv:2007.00808 [cs.IR] https://arxiv.org/abs/2007.00808

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 2369–2380. doi:10.18653/v1/D18-1259

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations (ICLR)*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models. *arXiv preprint arXiv:2506.05176* (2025).

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. 2024. WildChat: 1M ChatGPT Interaction Logs in the Wild. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=Bl8u7ZRlbM

Lucia Zheng, Neel Guha, Javokhir Arifov, Sarah Zhang, Michal Skreta, Christopher D. Manning, Peter Henderson, and Daniel E. Ho. 2025. A Reasoning-Focused Legal Retrieval Benchmark. In *Proceedings of the 2025 Symposium on Computer Science and Law* (Munich, Germany) *(CSLAW '25)*. Association for Computing Machinery, New York, NY, USA, 169–193. doi:10.1145/3709025.3712219

Yunfei Zhong, Jun Yang, Yixing Fan, Lixin Su, Maarten de Rijke, Ruqing Zhang, and Xueqi Cheng. 2026. Reason to Retrieve: Enhancing Query Understanding through Decomposition and Interpretation. arXiv:2509.06544 [cs.IR] https://arxiv.org/abs/2509.06544

A Extended Related Work

This section should be read as a less central extension of Section 2.

Large-scale retrieval evaluation has been shaped by benchmarks that progressively widened the notion of retrieval difficulty. MS MARCO (Bajaj et al., 2016) standardized passage ranking over real search queries and became the dominant supervision source for modern retrievers and rerankers. BEIR then broadened evaluation beyond in-domain ranking by assembling 18 datasets spanning multiple tasks and domains, and showed that zero-shot robustness remains a distinct problem: BM25 is often a surprisingly strong baseline, while late-interaction and reranking architectures tend to achieve stronger effectiveness at substantially higher cost (Thakur et al., 2021; Robertson and Zaragoza, 2009).

BRIGHT explicitly argues that many realistic retrieval queries require reasoning beyond lexical overlap or shallow semantic similarity (Su et al., 2025). A myriad of recent benchmarks extend this trend from "reasoning-intensive" to "complex-intent" retrieval (Wolfson et al., 2025; Thakur et al., 2025; Zheng et al., 2025). For example, CRUMB assembles eight diverse tasks with multi-part or constrained information needs and shows that even strong retrieval models remain far from saturated, while BrowseComp and BrowseComp-Plus evaluate deep-research agents on hard browsing or fixed-corpus search tasks that require iterative search and reasoning (Killingback and Zamani, 2025; Chen et al., 2025b; Wei et al., 2025).

Modern first-stage retrieval is dominated by architectures that compute document-side representations independently of the query. DPR popularized the supervised dual-encoder paradigm for open-domain

QA (Karpukhin et al., 2020); ANCE showed that realistic hard negatives from an approximate nearest-neighbor index substantially improve dense retriever training (Xiong et al., 2020); Contriever demonstrated that unsupervised contrastive pretraining can yield strong zero-shot dense retrieval (Izacard et al., 2022); and ColBERT introduced late interaction, preserving token-level matching signals while still precomputing document representations offline (Khattab and Zaharia, 2020).

A separate line of work tries to repair first-stage retrieval directly by changing the query rather than the scorer. EAR generates multiple expansions and reranks them before retrieval (Chuang et al., 2023). Query-rewriting work for retrieval-augmented LLMs similarly argues that the gap between user input and searchable evidence can often be narrowed by reformulating the query itself (Ma et al., 2023a). ReDI pushes this further with a reasoning-enhanced framework that decomposes complex queries into sub-queries, adds semantic interpretations, and fuses the results; notably, the paper reports gains on both BEIR and BRIGHT (Zhong et al., 2026). DIVER combines document processing, iterative query expansion, reasoning-enhanced retriever, and reranking for reasoning-intensive information (Sun et al., 2026).

Finally, multi-hop retrieval is adjacent but not identical. HotpotQA, 2WikiMultiHopQA, MuSiQue, and Hover were designed to require multi-document reasoning and supporting-evidence identification (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022; Jiang et al., 2020). Methods such as Multi-Hop Paragraph Retrieval and newer agentic or generate-then-ground pipelines address iterative evidence accumulation across documents (Feldman and El-Yaniv, 2019; Shi et al., 2024; Chang et al., 2026).

B Examples of the Five Tasks

Table 5 illustrates example queries, positive documents, and negative document for all five tasks in OBLIQ-Bench.

C Listwise Re-ranking Algorithm for GPT-5.2 Tournament

Given N candidates, we shuffle the pool and partition it into $\lceil N/b \rceil$ batches of size b. GPT-5.2 performs one listwise ranking call per batch, sorting all b candidates from most to least relevant. The top k candidates from each batch are promoted to the next round, and the remaining candidates are recorded as the tail for that recursion depth. This process repeats on the promoted candidates until the survivor pool fits in a single batch, which is ranked directly. The final ranking is formed by concatenating the final survivor ranking with the eliminated tails in reverse order of elimination depth. In our experiments, we set $b = 20$ and $k = 4$.

D Dataset Construction Prompts

Here we document all prompts used in the OBLIQ-BENCH construction pipeline. We organize prompts by benchmark and pipeline stage, following the shared construction framework described in §4 and illustrated in Figure 4. Each benchmark instantiates some or all of the five stages: (1) lens definition, (2) LLM annotation, (3) label collapse/clustering, (4) query generation, and (5) pool-and-expand evaluation.

D.1 Twitter-Conflict: Descriptive Retrieval over Implicit Stance

The Twitter-Conflict benchmark targets implicit political stance expressed through irony, hedging, and framing. The construction pipeline extracts implicit meanings, collapses semantic labels into canonical themes, generates queries that avoid source vocabulary, and performs multi-stage relevance verification.

D.1.1 Stage 2: Implicit Meaning Extraction

Each tweet labeled as `implicit` is processed to extract its underlying stance. GPT-5 receives batches of tweets and produces one-to-two sentence descriptions of the implied meaning.

Table 5: **Examples from OBLIQ-Bench.** Each row shows a query, a relevant document, and a negative document that were shortened for presentation here.

Query	Relevant document	Negative document
Twitter-Conflict		
Find tweets where users are implicitly linking recent run-ups in energy costs to overseas hostilities and seaborne transit bottlenecks...	@PetsatHome 24 tins of Chappie on the 25th January 2026 was £30.99. ... gone up to £37.79. Why so much? is it coming through the Strait of Hormuz?	Strait Of Hormuz Disruption Could Spike Global Prices Fast ... The economic risk is straightforward: oil and gas chokepoints drive inflation.
WildChat Errors		
Find conversations where the AI flip-flops about whether it can make pictures: at one point it says it can't produce image files... but elsewhere it claims to have created an image...	[User]: You can MAKE images [Assistant]: I can help create images based on your descriptions! ... I'll do my best to generate it for you. ... [User]: Make a realistic image of a gorilla with its torso replaced by a pencil in a big city at night on the street [Assistant]: I don't have the capability to create or generate images directly.	[User]: can you generate images [Assistant]: No, I can't generate images. However, I can help describe images, suggest ideas for them, or provide information about image-related topics.
Math Meta-Program		
An urn contains a white balls and b black balls... Drawing halts when three white balls are drawn in succession. Let X be the number of isolated pairs of white balls... and let Y be the number of isolated white balls.	A random walk starts at the origin and moves up-right or down-right with equal probability. What is the expected value of the first time that the walk is k steps below its then-current all-time high? ...	We have n balls, labeled 1 through n, and n urns, also labeled 1 through n... Let the random variable X be the number of balls that end up in the urn bearing their own number. Show that the expected value of X is $n - H_{n-1}$.
Writing-Style		
Every now and then, people ask me why I write. I don't get paid to write here, so it's not immediately obvious why I keep writing. ... You need to do the writing. Not AI. Writing is exercise.	There's a lot of noise about how AI is changing programming these days. It can be a bit overwhelming. ... The challenge with all of this is that coding agents really are performing some science fiction feats which were barely imaginable just 12 months ago.	People who have closely followed my work for the past few years have probably noticed that my output has gradually slowed. My last post on this blog was published over four years ago. ... My enthusiasm for my hobbies has always ebbed and flowed.
Congress Hearings		
There's this exchange from a Capitol Hill oversight session that I can't get out of my head. It wasn't about cryptocurrency or big tech — it was about the entities that police brokerages and trading firms... Then he turned to the executive running one of these hybrid oversight bodies and essentially forced him to define what he is — a businessman or a bureaucrat?	One of my favorite expressions over the years was when William O. Douglas was SEC Chairman. He said that self-discipline is always more welcome than discipline imposed from above. ... do you consider yourself a wholly private actor or a state actor with authority from the government when you think about your job as CEO?	By the 2000s, financial market regulators such as the SEC and FINRA were developing the capacity to collect and analyze raw data feeds directly from regulated entities. ... Instead of receiving periodic reports, those subscribing to FINRA's TRACE reporting system now have firehoses of real-time data to manage and analyze.

Implicit Meaning Extraction

System: You are an expert at analyzing implicit political discourse on Twitter/X.
For each tweet, extract the IMPLICIT MEANING—the stance, attitude, or position that is communicated through indirection (sarcasm, irony, hedging, selective framing, euphemism, or rhetorical questions) rather than stated directly.
Write a 1-2 sentence description of what the tweet is really saying or implying. Focus on the underlying stance, not the surface content.
User: [Batch of 15 tweets with IDs]

D.1.2 Stage 3: Label Collapse

Raw implicit meanings are collapsed into canonical themes through iterative consolidation.

Theme Assignment

System: You are clustering implicit political stances into thematic groups.
For each implicit meaning description, assign a short theme string (3-8 words) that captures the core stance. Similar stances should receive identical theme strings.
Be consistent: if two descriptions express the same underlying position (even with different wording), assign the same theme.
User: [Batch of 80 implicit meaning descriptions]

Theme Consolidation

System: You are consolidating a list of theme labels.
Many labels express the same underlying stance with slightly different wording. Collapse near-duplicates into canonical labels. Keep genuinely distinct stances separate.
Return a JSON mapping every original label to its canonical form.
Rules:
- Every input label must appear exactly once as a key
- Pick the clearest, most descriptive phrasing as canonical
- Multiple originals can map to the same canonical
- When uncertain whether two labels refer to the same stance, keep them separate

User: [Chunk of 150 theme labels]

D.1.3 Stage 4: Query Generation

For each canonical theme, a retrieval query is generated that captures the abstract stance without using vocabulary from the source tweets.

Query Generation with Relevance Grading

System: You are generating retrieval queries for an implicit stance benchmark.
You will receive a theme label and all tweets expressing that implicit stance.
Tasks:
1. Write ONE retrieval query (10-15 words) that a researcher would use to find tweets expressing this stance
2. The query must capture the ABSTRACT stance, not surface content
3. The query must NOT use words from the tweets verbatim
4. The query must NOT use named entities (people, places, organizations)
5. Grade each tweet's relevance: 2 = directly expresses this stance, 1 = tangentially related

User: Theme: [canonical theme]
Member tweets: [all tweets in cluster with implicit meanings]

D.1.4 Stage 5: Pool and Expand

Top results from each retriever are judged to expand relevance annotations.

Pooled Relevance Judgment (Twitter)

System: You are judging relevance for an information retrieval benchmark on political tweets.
You will receive:
- A RETRIEVAL QUERY (abstract, captures an implied stance)
- GOLD RELEVANT TWEETS confirmed relevant (for calibration)
- CANDIDATE TWEETS that are unjudged

For each candidate, judge relevance based on its IMPLICIT MEANING and stance, not surface keywords.
Relevance grades:
- 2 = clearly relevant: strongly matches the implied stance the query seeks
- 1 = marginally relevant: partially matches or tangentially related
- 0 = not relevant: different topic or stance

Respond with a JSON array: `[{"id": "<id>", "score": <0|1|2>}, ...]`

User: QUERY: [query]
GOLD TWEETS: [confirmed relevant tweets]
CANDIDATES: [unjudged tweets from retriever top-k]

D.1.5 Query Rewriting (Evaluation Baseline)

Query Rewriting for Implicit Discourse

System: You are helping build an information retrieval benchmark focused on IMPLICIT political discourse on Twitter/X.
The task: given a query that describes a political stance, retrieve tweets that EXPRESS that stance IMPLICITLY—through sarcasm, irony, hedging, euphemism, selective framing, understatement, or rhetorical questions. The relevant tweets will NOT state the position directly and will likely share little vocabulary with the query.
Example match direction:
Query: "Find tweets implicitly defending Tehran's strikes as legitimate retaliation against prior US-Israeli aggression"
Match: "So when you poke the bear for decades, you're surprised it bites?"
Rewrite the query into a form that will perform better with an embedding model:
- Capture the same underlying stance
- Phrase it closer to how the stance would be expressed implicitly on Twitter
- Remain a query (not a tweet), but bridge the gap between explicit description and implicit register
- Stay concise (1-3 sentences)
- Avoid verbose academic language

User: ORIGINAL QUERY: [query text]

D.2 WildChat: Descriptive Retrieval over LLM Failure Modes

The WildChat benchmark targets conversations exhibiting specific LLM failure modes (format violations, silently dropped instructions, unjustified transformations) where the failure has no lexical marker in the text.

D.2.1 Stage 2: Failure Mode Annotation

GPT-5.4-nano annotates all conversations to identify behavioral failures.

Failure Mode Sweep

System: You are an expert at identifying LLM failure modes in human-AI conversations.
Analyze this conversation and identify any behavioral failure where the AI:
- Violates a formatting constraint the user specified
- Silently drops or ignores part of the user's instruction
- Makes an unjustified transformation (unit conversion, format change, etc.)
- Fails to self-correct after producing incorrect output
- Exhibits any other systematic deviation from the user's request

If a failure is present, return:
- `failure_type`: short label (3-8 words)
- `description`: one sentence describing the specific failure

If no failure, return `null`.
User: [Conversation transcript]

D.2.2 Stage 3: Label Collapse

Raw failure labels are collapsed into canonical types through embedding-based clustering and LLM consolidation.

Conservative Label Collapse

System: You are cleaning up a list of LLM failure mode labels.
Many labels express the same underlying mistake type with slightly different wording. Collapse near-duplicates ONLY when they clearly describe the same failure mode. When in doubt, keep labels SEPARATE—this pass should be conservative. Prefer splitting over merging for anything ambiguous.
Return a JSON object mapping every original label to its canonical form.
Rules:
- Every input label must appear as a key
- Pick the clearest, most descriptive phrasing as canonical
- When uncertain whether two labels refer to the same failure, map each to itself

User: [Cluster of similar failure labels]

Distinctness Check Against Existing Queries

System: You are checking whether a CANDIDATE LLM-failure-mode retrieval query would duplicate any existing query in a retrieval benchmark.
You'll receive:
- A CANDIDATE mistake type (short label + sample descriptions)
- A list of EXISTING retrieval queries already in the benchmark

Decide whether the candidate describes a failure mode SUBSTANTIVELY THE SAME as any existing query. Two queries are the same if a researcher would expect the same conversations to match both, even if worded differently.
Err on the side of DUPLICATE. If the candidate is a more specific subtype of an existing query, call it a duplicate. Only mark as distinct when the candidate clearly describes a different causal mechanism or surface behavior.
User: CANDIDATE: [failure type with descriptions]
EXISTING QUERIES: [list of current benchmark queries]

D.2.3 Stage 4: Query Generation

Novel Query Generation

System: You are extending a hard information retrieval benchmark for LLM failure mode analysis.
You'll receive:
- A group of AI-human conversations sharing one failure mode
- A list of EXISTING queries already in the benchmark

Tasks:
1. Write one NEW retrieval query a researcher might use to find these conversations
   - Natural phrasing: "Find conversations where the AI..."
   - Must capture the ABSTRACT failure pattern, not surface content
   - Must NOT use words from the descriptions verbatim
   - Must be discriminative: specific enough to exclude unrelated failures
   - Must NOT overlap with any existing query's failure mode
2. If you produced a query, grade each conversation's relevance (2 = central, 1 = tangential)

If you cannot write a query that is clearly distinct from all existing queries, set `query` to `null`. Dropping a near-duplicate is correct.
User: FAILURE TYPE: [canonical label]
CONVERSATIONS: [member conversations with descriptions]
EXISTING QUERIES: [current benchmark queries]

D.2.4 Stage 5: Pool and Expand

Pooled Relevance Judgment (WildChat)

System: You are an expert at identifying LLM failure modes in human-AI conversations.
You will receive a QUERY describing a very specific LLM failure pattern, and a set of CANDIDATE CONVERSATIONS that have not yet been judged for relevance.
A candidate is relevant ONLY IF:
1. The user's instruction matches the type of constraint described in the query
2. The AI's response violates that constraint in the specific way the query describes
3. A reasonable person would agree the failure is the same, not merely analogous
Additional guidelines:
- A candidate can be relevant even if the deviation appears unintentional or minor—what matters is whether the output differs from the exact specification
- When instructions contain errors, judge against what the user actually specified

A candidate is NOT relevant if:
- The AI makes a different kind of mistake
- The failure mechanism differs from what the query describes
- The AI actually complies correctly

Scoring: 2 = clearly relevant, 1 = marginally relevant, 0 = not relevant
Be conservative. When in doubt, assign 0.
User: QUERY: [query]
CONVERSATIONS TO JUDGE: [candidate conversations]

D.3 Math Meta-Program: Analogues via Shared Reasoning Technique

The Math Meta-Program benchmark targets problems sharing the same abstract proof strategy across completely different mathematical fields.

D.3.1 Stage 2: Fingerprinting

GPT-5 analyzes each problem and its solution to extract a reasoning fingerprint.

Reasoning Fingerprint Extraction

System: You are a mathematics professor who has spent decades studying how mathematical reasoning recurs across competitions, fields, and difficulty levels.
Your task: read a problem and its solution(s), then extract a REASONING FINGERPRINT that captures the abstract cognitive move required—the "aha moment"—stated so domain-independently that a professor would recognize the same move in a problem from a completely different field.
CRITICAL: GRANULARITY OF meta_strategy AND fingerprint_summary
These fields are the PRIMARY CLUSTERING KEYS. They must be abstract enough that problems from completely different fields using the same reasoning move land in the same cluster.
RULE: The fingerprint_summary must be abstract enough to match a family of problems, specific enough to exclude unrelated reasoning moves.
TOO COARSE (useless): "induction", "pigeonhole", "contradiction"
TOO FINE (useless): anything that describes exactly one problem
Output fields:
- `meta_strategy`: The abstract reasoning move, the eureka. Use NO domain vocabulary. Must describe the logical maneuver so that a problem from a different field using the same move would fit.
- `abstract_proof_move`: Domain-independent logical skeleton. Examples: "sum over all rotations $\rightarrow$ global average $\rightarrow$ existence by integrality" | "assume extremal $\rightarrow$ local exchange argument $\rightarrow$ contradiction"
- `key_insight`: One sentence: the non-obvious observation that unlocks the problem. State WITHOUT domain vocabulary.

- `fingerprint_summary`: $\leq$20 words. The meta-program label for this family. PRIMARY CLUSTERING KEY.
- `technique_family`: algebra | combinatorics | geometry | number_theory | real_analysis | linear_algebra | probability | game_theory
- `difficulty_tier`: easy | medium | hard

User: PROBLEM: [problem statement]
SOLUTION: [solution text]

D.3.2 Stage 3: Clustering

Meta-Program Label Normalization

System: You are normalizing a list of meta-program labels from mathematical reasoning analysis.
Many labels express the same underlying reasoning move with slightly different wording. Collapse near-duplicates into a single canonical label. Keep genuinely distinct reasoning moves as separate canonicals.
CRITICAL DISTINCTION: Two labels should collapse only if a mathematician would say "yes, these require the same core aha moment"—NOT merely that they use the same technique family, topic area, or proof technique.
Rules:
- Every input label must appear exactly once as a key
- Pick the most abstract, clearly-phrased version as canonical
- Do NOT collapse labels that are merely in the same technique family

User: [List of meta-program labels to normalize]

Cluster Merge Identification

System: You are an expert mathematician reviewing clusters of math problems grouped by their meta-program—the abstract reasoning move required to solve them.
You will receive a list of clusters, each with a canonical label and sample fingerprint_summaries of member problems.
Identify which clusters should be MERGED because they represent the same fundamental reasoning pattern—the same abstract "aha moment"—even if described differently.
DO NOT merge clusters that merely belong to the same technique family (e.g., "all use induction") but require genuinely different insights.
Return merges as: `{"merges": [{"merge_ids": [...], "new_label": "...", "rationale": "..."}]}`

User: [List of clusters with labels and sample fingerprints]

D.3.3 Stage 3b: Cluster Validation

Cluster Membership Verification

System: You are an expert mathematician who specializes in recognizing when problems require the same abstract reasoning pattern—the same "aha moment"—even across completely different fields.
You will receive a group of problems automatically clustered as sharing the same meta-program. Your job is to verify which ones truly belong.
A problem belongs in the cluster if and only if:
- Its core proof insight is structurally the same as the majority of other members
- A student who mastered the pattern would immediately recognize the connection, even if topics look completely different

Problems in the same cluster CAN be from varying mathematical fields—it's the reasoning style that matters.
A problem does NOT belong if:
- It uses a similar technique family but the specific aha moment is different

- Its fingerprint was accidentally grouped due to similar wording but the underlying reasoning move is distinct

Flag for removal only problems where you are confident the core insight is genuinely different. When in doubt, keep.
Return: `{"keep": [...], "remove_candidates": [...], "canonical_label": "...", "rationale": "..."}`
User: CLUSTER LABEL: [label]
MEMBER PROBLEMS: [problems with fingerprints]

D.3.4 Cluster Selection

Diversity-Based Cluster Selection

System: You are helping curate a math reasoning-analogue retrieval benchmark.
You will receive descriptions of clusters of math problems. Each cluster groups problems sharing the same abstract reasoning pattern regardless of field.
Task: Greedily select clusters that maximize diversity.
WITHIN-CLUSTER SURFACE DIVERSITY (makes individual queries hard): A cluster is STRONG if its members span DIFFERENT mathematical fields (geometry, number theory, analysis, etc.). This forces retrievers to understand the abstract reasoning pattern, not match surface keywords.
Each cluster shows within-diversity = HIGH (3+ fields) / MED (2) / LOW (1). HIGH and MED clusters should make up the bulk of your selection.
STOPPING RULE: Stop when all remaining candidates have LOW within-diversity.
User: [Cluster descriptors with diversity ratings]

D.3.5 Stage 5: Pool and Expand

Pooled Relevance Judgment (Math)

System: You are judging relevance for a mathematical reasoning-analogue retrieval benchmark.
You will receive:
- A QUERY PROBLEM and its reasoning fingerprint
- GOLD PROBLEMS confirmed to share its meta-program (for calibration)
- CANDIDATE PROBLEMS that are unjudged

For each candidate, decide if it shares the same CORE REASONING PATTERN—the same abstract "aha moment"—regardless of topic, vocabulary, or mathematical field.
Relevance grades:
- 2 = SAME meta-program: a mathematician would immediately recognize the same eureka insight across different fields. Abstract proof move is structurally identical.
- 1 = ADJACENT reasoning: genuinely related proof strategy but a distinct core insight. A student who mastered the query would be helped but not immediately equipped.
- 0 = DIFFERENT reasoning pattern entirely.

User: QUERY: [problem with fingerprint]
GOLD: [confirmed analogues]
CANDIDATES: [problems to judge]

D.3.6 Evaluation: Listwise Ranking

MergeSort-Interleave Ranking with no solutions

System: You are an expert mathematician. Your task is to identify problems that share the same ABSTRACT PROOF STRATEGY, i.e., the same "meta-move", even when topics look completely different.
IMPORTANT: You must INFER how each problem would be solved. You don't have access to solutions—you must figure out the likely proof approach from the statement alone.
WHAT "SAME ABSTRACT STRATEGY" MEANS:

Two problems match if their solutions would use the same high-level meta-move:
• "Exploit uniqueness of X to constrain the answer"
• "Transform representation → swap order of operations → collapse"
• "Diagonalize/decompose → solve per component → reassemble"
• "Finite domain: injective ⇔ surjective"
• "Symmetry averaging to project onto invariants"
Problems can share a meta-move despite having NOTHING in common on the surface.
IGNORE surface similarity: same field, same objects, similar notation, problems that "look alike"
FOCUS ON deep proof skeleton: What KEY INSIGHT unlocks the problem? What meta-level principle does the solution rely on? Would the same proof OUTLINE work for both?
Rank ALL candidates. Return: `{"ranked": [{"candidate_num": N, "reason": "..."}, ...]}`
User: QUERY: [problem statement]
CANDIDATES: [numbered list of problems]

MergeSort-Interleave Ranking with solutions

System: You are an expert mathematician. Your task is to identify problems that share the same ABSTRACT PROOF STRATEGY, i.e., the same "meta-move".
You are given BOTH problem statements AND solutions. Use the solutions to identify the core reasoning technique, not just surface similarity.
IMPORTANT: Mathematical problems often have MULTIPLE valid solution approaches. Two problems may share a reasoning pattern via one proof method even if other solutions exist. Consider ALL plausible strategies:
• A counting problem might use bijection, generating functions, or recursion
• A number theory problem might use modular arithmetic, induction, or construction
• An inequality might yield to AM-GM, Cauchy-Schwarz, or calculus
If ANY natural solution approach shares the same core insight, treat them as matching.
FOCUS ON the deep proof skeleton revealed in solutions: What KEY INSIGHT unlocks each? What meta-level principle does each rely on? Do solutions follow the same structural pattern?
User: QUERY: [problem with solution excerpt]
CANDIDATES: [problems with solution excerpts]

Two-Stage Think-First Ranking

System: You are an expert mathematician identifying problems with shared abstract proof strategy.
IMPORTANT: Work in TWO STAGES:
STAGE 1 - THINK FIRST: Before ranking, reason through how each problem would be solved. For the query and each candidate: What's the key insight? What proof technique? What's the abstract meta-move?
STAGE 2 - RANK: Only after thinking through each problem's approach should you rank by whether they share the query's abstract proof strategy.
Return:

```
{
  "query_analysis": {"key_insight": "...", "meta_move": "..."},
  "candidate_analyses": [{"candidate_num": N, "key_insight":

  "...", "meta_move": "..."}, ...],
  "ranked": [{"candidate_num": N, "reason": "..."}, ...]
}
```

User: QUERY: [problem]
CANDIDATES: [numbered problems]

D.4 Writing-Style: Analogues via Cross-Domain Authorship

The Writing-Style benchmark targets authorship identification across unrelated topics, testing whether stylistic invariants persist when topic is factored out. Unlike other benchmarks, Writing-Style skips Stages 2–3 (authorship is ground truth) and proceeds directly to evaluation.

D.4.1 Evaluation: Listwise Ranking

MergeSort-Interleave Ranking (Authorship)

System: You are an expert in authorship analysis and stylometry.
You will receive a QUERY SNIPPET and CANDIDATE SNIPPETS. Rank candidates by how likely they were written by the same author as the query.
Focus on writing style, NOT topic:
- Sentence rhythm and syntactic patterns
- Hedging language and epistemic stance
- Characteristic vocabulary and phrasing
- Tone and rhetorical posture

The same author may write about different topics—topic similarity is NOT the signal you should rely on.
Every candidate number MUST appear exactly once.
Return: `{"ranked": [{"candidate_num": N, "reason": "<brief style observation>"}, ...]}`
User: QUERY SNIPPET:
"""[query text]"""
CANDIDATE SNIPPETS: [numbered snippets]
Rank by stylistic similarity to the query author.

D.4.2 Evaluation: Multi-Hop Retrieval

Multi-Hop Query Generation (Authorship)

System: You are an expert at author attribution, i.e, identifying texts written by the same author. Given an original text snippet and notes from previous search iterations, generate a search query that will help find other texts written by the same author.
The search query should:
1. Focus on distinctive stylistic features, vocabulary patterns, or thematic preferences
2. Capture the author's unique voice and writing mannerisms
3. Be different from previous search angles to maximize coverage

If this is the first hop, focus on the most distinctive stylistic markers. For later hops, refine based on patterns you've discovered.
User: ORIGINAL TEXT: [query snippet]
NOTES FROM PREVIOUS HOPS: [accumulated observations]
HOP NUMBER: [N] of [total]

Multi-Hop Note Extraction (Authorship)

System: You are an expert at author attribution.
Your tasks:
1. Select text snippets that appear to be written by the SAME AUTHOR as the query
2. Write brief notes about the stylistic patterns you observed

Look for: vocabulary choices, sentence structure, punctuation habits, thematic preferences, tone, rhetorical devices, and other authorial fingerprints.
User: QUERY TEXT: [snippet]
PREVIOUS NOTES: [observations]
CANDIDATES: [retrieved snippets]
Return: `{"candidate_ids": [...], "notes": "...", "summary": "..."}`

Query Rewriting (Authorship)

System: You are an expert at author attribution.
Given a text snippet, rewrite the query to better capture the distinctive stylistic features that would help find other texts by the same author. Focus on vocabulary patterns, sentence structure, tone, and other authorial fingerprints.
Return: `{"rewritten_query": "...", "rationale": "..."}`
User: ORIGINAL TEXT: [snippet]
Rewrite to capture the author's distinctive style.

D.5 Congress Hearings: Tip-of-the-Tongue Retrieval

The Congress Hearings benchmark targets tip-of-the-tongue retrieval: matching fuzzy, impressionistic recollections to specific hearing passages. Unlike other benchmarks, Congress skips Stage 3 (each query targets a single passage rather than a cluster).

D.5.1 Stage 2: Memorability Rating and Query Generation

ToT Query Generation from Memorable Passages

System: You are helping construct a "Tip of the Tongue" (ToT) information retrieval benchmark over US congressional hearing transcripts.
You will be shown a passage from a hearing—typically a Q&A exchange between a legislator and a tech industry witness.
Your job:
1. RATE the passage's memorability (1-5):
- 1 = boring procedural, nobody would remember
- 2 = mildly interesting but generic
- 3 = somewhat memorable, has a specific detail worth recalling
- 4 = very memorable, a distinct confrontation or revelation
- 5 = iconic, widely reported moment

2. If memorability $\geq$ 3, write a ToT POST ($\sim$200 words, written as someone posting on Reddit trying to recall this moment):
MUST FOLLOW:
- Do NOT include names of any person, company, platform, committee, or legislation
- Do NOT include dates, years, or exact identifiers
- Reflect imperfect memory: mix up minor details, be uncertain about specifics, conflate with adjacent moments. Do NOT explicitly say "I'm not sure"—state distorted details naturally
- Describe what happened in indirect, experiential terms—what it felt like, the dynamic, body language, tone
- Natural conversational tone. No formal writing. Write like you're posting on Reddit
- Skip greetings like "Hey everyone"—begin directly with your memory
- End with an open-ended question inviting others to help find it
- The post must be genuinely hard for search to match. Avoid verbatim words from the passage

COULD FOLLOW:
- Set the scene with a personal anecdote about when/where you encountered this
- Focus on emotional impact—tension, awkward silence, audible reaction
- Make subtle comparisons to similar moments without naming them
- Include 1-2 plausible but INCORRECT details (memory distortion)
- Mention sensory details: camera zoom, sounds from gallery, fidgeting
- Reference the emotional arc: buildup, confrontation, deflection

BAD (keywords overlap, too specific):
"When did the Facebook CEO testify about data privacy before the Senate?"
GOOD ($\sim$200 words, oblique, personal, distorted):
"So I was procrastinating at work a few years back and ended up watching one of those government hearings where they bring in tech people, and there was this one exchange that's stuck with me. One of the older guys on the panel—I want to say he was from somewhere in the south but I might be mixing him up—he pulled out what looked like a printout of something from inside the

company itself. Like internal stuff. And he read part of it out loud, and the exec at the table just had this deer-in-headlights moment..."
User: PASSAGE: [hearing transcript excerpt]

D.5.2 Opening Style Diversification

To prevent mode collapse in query formulation, we diversify opening styles using 15 distinct patterns:

Opening Diversification

System: You are rewriting forum posts to have more diverse openings.
You will receive a ∼200 word forum post where someone is trying to recall a congressional hearing moment. The content and meaning must stay EXACTLY the same. Your ONLY job is to rewrite the opening 1-2 sentences to use this specific style:
STYLE: [one of the styles below]
Rules:
- Keep the EXACT same content, details, memory distortions, and closing question
- ONLY change the first 1-2 sentences to match the requested style
- Do NOT add or remove any information
- Output ONLY the rewritten post

Opening styles used:
1. Frustrated question ("This has been driving me crazy...")
2. Mid-thought, no preamble ("Ok so there was this hearing where...")
3. Setting a scene ("I was at my desk / on the couch...")
4. A comparison ("It was kind of like that other time when...")
5. Challenge to the reader ("Does anyone else remember...")
6. Stating what stuck ("The thing that always stuck with me was...")
7. Diving straight in ("There's this clip where...")
8. Temporal anchoring ("A couple years back, maybe around election season...")
9. Explaining why you're posting ("My coworker mentioned something today...")
10. Emotional reaction first ("I still get secondhand embarrassment...")
11. A disclaimer ("I might be mixing up two different things here but...")
12. Referring to how you saw it ("Someone sent me a clip once of...")
13. Strong opinion opener ("Honestly one of the wildest moments...")
14. A question to yourself ("Why can I never find this clip again?")
15. Anchoring to another memory ("So right around the time that scandal...")

D.5.3 Evaluation: Query Rewriting

ToT Query Rewriting for Transcript Matching

System: You are an advanced retrieval system. You will be given a tip of tongue query describing a user's hazy memory of a specific moment from a US congressional hearing. They wrote a vague description of what they remember.
Rewrite their description as a search query that is more likely to match the actual hearing transcript. Use the kind of language that would appear in an official transcript based on whatever you can infer from their description.
Return ONLY a JSON object:

```
{
  "rewritten_query": "<your rewritten query>",
  "reasoning": "<what you think they're describing and why>"
}
```

User: {query}
Rewrite this to better match the actual hearing transcript.

D.5.4 Evaluation: Multi-Hop Retrieval

Multi-Hop Query Generation (Congress ToT)

System: You are an advanced retrieval system. You will be given a tip of tongue query describing a user's hazy memory of a specific moment from a US congressional hearing. They wrote a vague description of what they remember. Given their description and any notes from previous search attempts, generate a search query that would match the actual hearing transcript.
Think about what the transcript would actually contain—speaker names, committee procedures, policy terms, specific phrases.
User: ORIGINAL DESCRIPTION:
{original_query}
NOTES FROM PREVIOUS HOPS:
{notes}
HOP: {hop_num} of {total_hops}
Generate a search query. Return ONLY JSON:

```
{
  "search_query": "<query using transcript-language>",
  "rationale": "<what you're looking for this hop>"
}
```

Multi-Hop Note Extraction (Congress ToT)

System: You are trying to find a specific congressional hearing moment that someone vaguely remembers. You will see their description and some candidate transcript passages.
Select any passages that could plausibly be the moment they're describing. Write notes about what you've learned so far—what matches, what doesn't, what to search for next.
User: WHAT THEY REMEMBER:
{query}
PREVIOUS NOTES:
{previous_notes}
CANDIDATE PASSAGES (hop {hop_num}):
{candidates}
Analyze these passages. Return ONLY JSON:

```
{
  "candidate_ids": ["id1", "id2", ...],
  "notes": "<what matched, what didn't, what to try next>",
  "summary": "<brief summary of this hop>"
}
```

{selection_instruction}

Multi-Hop Reranking (Congress ToT)

System: You are trying to find a specific congressional hearing moment. Rank passages by how likely they are the moment being described.
Output exactly {k} items. Every candidate number must appear exactly once.
Return ONLY JSON:

```
{
  "ranked": [
    {"candidate_num": <int>, "reason": "<brief>"},
    ...
  ]
}
```

User: WHAT THEY REMEMBER:
{query}
CANDIDATE PASSAGES:
{candidates}

Rank all {k} candidates. Every number must appear exactly once.