SUPERINTELLIGENT RETRIEVAL AGENT: THE NEXT FRONTIER OF AGENTIC RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Retrieval-augmented agents are increasingly the interface to large organizational and public knowledge bases, yet most still treat retrieval as a black box: they issue exploratory queries, inspect returned snippets, and iteratively reformulate until useful evidence emerges. This resembles how a newcomer searches an unfamiliar database rather than how an expert navigates it with strong priors about terminology, constraints, and likely evidence, leading to unnecessary retrieval rounds, increased latency, and poor recall.

We introduce *Superintelligent Retrieval Agent* (SIRA), which defines *superintelligence* in retrieval as the ability to compress multi-round exploratory search into a single corpus-discriminative retrieval action. SIRA does not merely ask what terms are relevant to the query; it asks which terms are likely to separate the desired evidence from corpus-level confusers. On the corpus side, an LLM enriches each document offline with missing search vocabulary; on the query side, it predicts evidence vocabulary omitted by the query; and corpus statistics are used as tool calls to filter proposed terms that are absent, overly common, or unlikely to create retrieval margin. In the main BEIR and QA setting, the final retrieval step is a single weighted BM25 call combining the original query with the validated expansion.

Across ten BEIR benchmarks, SIRA achieves the strongest average retrieval performance in our comparison, outperforming dense retrievers, learned sparse retrievers, and LLM-based search-agent baselines while using no relevance labels or retriever fine-tuning. On question answering, SIRA's retrieval-only answer coverage exceeds recent RL-trained agentic QA systems on NQ and HotpotQA. Finally, we introduce **BrowseComp-Wikipedia**, a hard-search benchmark of 232 BrowseComp-derived queries grounded in a 25,587,229-document English Wikipedia index. Even without index-time LLM document enrichment, using only grounded Wikipedia categories as corpus-visible structure, SIRA outperforms multi-round Perplexity agents at every retrieval budget, reaching 9.70% Recall@1, 15.27% Recall@10, and 36.14% Recall@100. These results show that pre-planned, corpus-grounded lexical retrieval can outperform substantially more expensive multi-round search while remaining interpretable, training-free, and efficient.

1 INTRODUCTION

Information retrieval (IR) has evolved from lexical matching, exemplified by BM25 (Robertson & Zaragoza, 2009), to neural retrieval dominated by dense embeddings (Karpukhin et al., 2020). Embedding-based retrievers perform well when trained with abundant in-domain supervision: large-scale relevance labels and interaction logs that calibrate the model to a platform's user population (Bajaj et al., 2016; Joachims et al., 2007; Chapelle & Zhang, 2009). This regime has fueled modern *retrieval-augmented generation* (RAG) systems that ground LLM outputs in external corpora (Lewis et al., 2020). The user interface for information access is changing rapidly: search is increasingly *answer-forward* and *conversational*, with LLMs mediating multi-turn information seeking (Mo et al., 2025). A key consequence is that the classic training signal for supervised ranking, clickthrough, becomes sparse, delayed, and biased: users often terminate sessions without clicking, or accept an on-page summary as the final answer. A large-scale browsing analysis by Pew Research Center finds that when Google presents an AI-generated summary, users click standard result links substantially

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

less often (Pew Research Center, 2025). Click-based supervision is therefore becoming unreliable at scale, precisely as query behavior is shifting.

Compositional queries need controllable retrieval. At the same time, *query distributions* are moving away from short keyword strings toward longer, compositional requests that combine constraints, exclusions, and multi-step intent. This shift is a hallmark of conversational search. Pure similarity search is an awkward fit for this regime: dense retrieval exposes only a black-box nearest-neighbor operator and provides weak handles for enforcing structure (e.g., must-include/must-not-include constraints, attribute filters, or explicit decomposition). Neural sparse methods such as SPLADE (Formal et al., 2021) partially restore lexical controllability while preserving learning-based ranking, but they are still used as fixed retrievers inside pipelines rather than as controllable components of an agent policy. Classical lexical retrieval, exemplified by BM25, possesses underappreciated strengths that become decisive when paired with LLM reasoning. BM25 is *transparent*: an agent can boost keywords, enforce constraints, and decompose queries with predictable effects on retrieval outcomes. It naturally rewards *rare, discriminative terms* via IDF weighting, so domain-specific jargon that would be diluted in a dense embedding becomes a powerful retrieval signal. It is *auditable*: one can trace exactly which keywords matched and why, while avoiding the latency and memory costs of dense indices. The missing ingredient has been a mechanism to surface the right rare terms and constraints; LLMs, with their vast parametric knowledge, are uniquely positioned to fill this role.

LLM reasoning meets retrieval. The limitations of dense retrieval are not merely engineering inconveniences; they expose a deeper mismatch between single-vector representations and compositional information needs. Recent theoretical and empirical work shows that fixed-dimensional embeddings can realize only a limited family of relevance patterns (Weller et al., 2026), that static embeddings are information bottlenecks whenever relevance requires cross-attention-style interaction (Shrivastava, 2025), and that vector databases impose substantial cost, latency, and objective-mismatch burdens in production (Shrivastava, 2023) In parallel, LLM reasoning frameworks such as Chain-of-Thought, Tree-of-Thoughts, and Graph-of-Thoughts demonstrate that LLMs can plan and explore structured intermediate states (Wei et al., 2022; Yao et al., 2023; Besta et al., 2024). Tool-using agents extend this capability to external actions, including search (Yao et al., 2022), and recent reinforcement-learning methods train LLMs to interleave reasoning with multi-turn web search (Jin et al., 2025). In most agentic search systems, however, retrieval itself remains an opaque tool: the agent can rewrite queries and judge snippets, but cannot directly manipulate retrieval primitives such as keyword weighting, constraint selection, or decomposition tied to index-time signals.

This exposes why current search agents remain brittle despite strong reasoning capabilities. Agentic systems such as ReAct, IRCoT, and recent RL-trained search agents improve performance by interleaving reasoning with repeated retrieval calls (Yao et al., 2022; Trivedi et al., 2023; Jin et al., 2025). However, this success is partly obtained through a *retrieval-context advantage*: after each search, the agent absorbs returned snippets, discovered entities, surface vocabulary, and near misses into the LLM context, then uses this accumulated evidence to formulate later queries. In other words, the agent compensates for weak retrieval control by learning the corpus through interaction. This strategy is expensive and noisy, and it relies increasingly on long-context LLMs to retain and use many intermediate passages, a regime known to be unreliable when relevant evidence is buried in long contexts (Liu et al., 2024). Thus the failure mode is not simply that LLMs cannot reason; it is that the search interface gives them too little direct control, forcing them into hit-and-miss exploration.

The known hardness of retrieval that LLM agents are clueless about. Classical IR theory clarifies the missing ingredient. Retrieval is not just a question of whether a query is semantically related to the desired document; it is a comparative ranking problem in which the gold evidence must outrank many non-gold *confusers*. The probability ranking principle and learning-to-rank methods formalize this as ordering relevant documents above non-relevant alternatives (Robertson, 1977; Joachims et al., 2007; Chapelle & Zhang, 2009), while BM25 operationalizes corpus contrast through document frequency and IDF (Robertson & Zaragoza, 2009). A query can therefore be plausible in isolation yet fail because its terms also match many distractors, or because its most expert-sounding terms are absent, too common, or weakly discriminative in the target index. Dense single-vector retrieval adds a further bottleneck: recent theory shows that fixed-dimensional embedding retrievers cannot realize all top-k relevance patterns and can fail even on simple realistic constraint structures (Weller et al., 2026). This points to the core gap SIRA addresses: an LLM may know what relevant evidence should look like, but it needs index-visible statistics and explicit retrieval controls to make that expectation discriminative against the confusing articles that corpus may be flooded with.

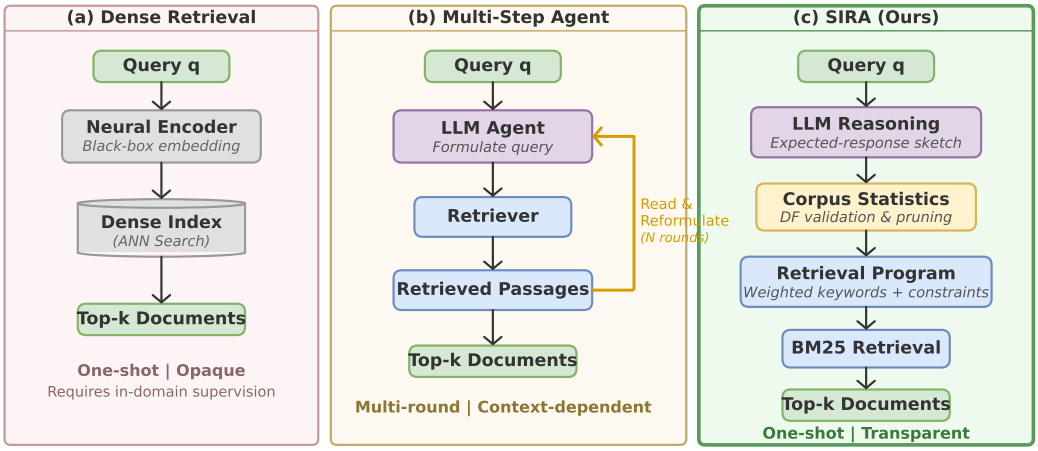


Figure 1: Three retrieval paradigms compared. **(a)** Dense retrieval encodes queries and documents into a shared embedding space and performs nearest-neighbor search; **(b)** Multi-step agent retrieval uses an LLM to iteratively formulate queries, read retrieved passages, and reformulate over N rounds; later queries benefit from accumulated retrieval context. **(c)** SIRA produces an expert-level retrieval action in a single shot: the LLM generates an expected-response sketch, validates proposed terms against corpus statistics, and compiles a controlled BM25 query with keywords.

SIRA formalizes the setting of one-shot, controllable BM25 retrieval across broad IR and QA benchmarks, building on recent evidence that LLM agents can generate discriminative keyword and grep queries, approaching RAG-level QA performance (Subramanian et al., 2025; Wang et al., 2026). Existing results, however, are either practitioner-level, code-centric, or multi-turn and context-accumulating (Karpathy, 2026; Pan et al., 2025). SIRA targets the single-query regime. A particularly important stress test is hard web-style search. BrowseComp was introduced to measure whether browsing agents can locate hard-to-find, short-answer facts on the web (Wei et al., 2025). Its questions are valuable because they are difficult to find but easy to verify, but the original benchmark relies on live web search, where the corpus, ranking function, and snippets are uncontrolled. BrowseComp-Plus improves reproducibility by constructing a fixed corpus of 100,195 documents for 830 verified queries (Chen et al., 2025). However, a curated 10^5-document corpus is still much smaller than the search spaces faced by real agents, and it weakens the core disambiguation problem: ranking the right page above millions of plausible confusers. We therefore introduce BrowseComp-Wikipedia, a 232-query benchmark derived from BrowseComp and grounded entirely in a 25,587,229-document English Wikipedia index. This setting preserves the hard-search character of BrowseComp while making retrieval controlled, public-corpus grounded, and substantially larger scale.

Goal. The central limitation of today's LLM-driven retrieval agents is that retrieval remains a *black-box environment* they explore through repeated interaction. Existing agents issue a query, inspect returned snippets, and reformulate using accumulated evidence, a *retrieval-context advantage* where later searches are conditioned on information exposed by earlier ones. This resembles a domain newcomer learning an unfamiliar database through exploration, rather than an expert who anticipates relevant terminology and evidence patterns before reading retrieved passages, as illustrated in Fig. 1. We define *superintelligence in retrieval* as the ability to replace this multi-round process with a single expert-level retrieval action: (i) form a domain-informed expectation of what relevant evidence looks like, (ii) ground that expectation using lightweight index-aware signals, (iii) compile the result into explicit retrieval controls, and (iv) execute retrieval efficiently and transparently.

Our proposal. We propose the *Superintelligent Retrieval Agent* (SIRA), a retrieval-centric agent that searches like a domain expert. Modern LLMs encode substantial parametric knowledge such as terminology, entities, relations, but lack a mechanism to convert this knowledge into precise retrieval actions over a target corpus. SIRA provides that mechanism through a scalable two-stage framework.

First, the LLM produces an *expected-response sketch*: a compact hypothesis of the concepts, entities, and discriminative terms likely to appear in relevant evidence. This sketch acts as a retrieval prior, not

as evidence. Before issuing the final BM25 query, SIRA consults lightweight corpus-statistics tools to validate and prune proposed terms without returning answer passages, avoiding the retrieval-context advantage enjoyed by multi-round agents. Second, conditioned on the sketch and index statistics, SIRA compiles a *retrieval program*: a single controlled BM25 query with weighted keywords, optional exclusions, and structured composition. We evaluate SIRA in this strict one-shot setting: one LLM reasoning step, optional index-statistic checks, and one BM25 call. This strict one-query setting is used for the BEIR and downstream QA experiments; Section 3.4 studies a budgeted multi-query extension for BrowseComp-Wikipedia. Multiple queries could improve performance, but the one-query regime isolates the central question: how far can retrieval go when the agent must formulate the right lexical action without reading retrieved snippets? BM25's reliance on exact matching becomes a strength when the LLM supplies the right vocabulary and verifies its discriminative value through corpus statistics.

We evaluate SIRA in three increasingly demanding settings. First, across ten BEIR-style retrieval benchmarks (Thakur et al., 2021), SIRA outperforms strong dense retrievers, learned sparse retrievers, and LLM-based retrieval-agent baselines on average Recall@10 and NDCG@10. Second, on downstream QA, SIRA's retrieval-only answer coverage exceeds recent RL-trained agentic QA systems on NQ and HotpotQA, even though those systems use trained search policies and answer generators. Third, on BrowseComp-Wikipedia, SIRA directly challenges the prevailing multi-round search-agent paradigm: pre-planned corpus-grounded BM25 queries, constructed without conditioning on retrieved page content, outperform Perplexity-based agents that can browse iteratively for up to 100 turns. These results suggest that the bottleneck in retrieval-augmented agents is not the sophistication of the reader or the number of search iterations, but the agent's ability to construct an expert-level, corpus-discriminative retrieval action.

2 SIRA: Superintelligent Retrieval Agent

2.1 Overview

Most retrieval-augmented agents interact with a search engine through a loop: issue a query, inspect results, reformulate, and repeat until useful evidence emerges. SIRA replaces this multi-round loop with a *one-shot* pipeline that bridges the vocabulary gap between queries and documents from *both sides* simultaneously. We use Lucene BM25 as the retrieval engine; Appendix A reviews the scoring formula and the index-visible DF/IDF signals used for term validation. The full system requires no training, no relevance labels, and no supervised query–document pairs; it operates with a frozen LLM and corpus statistics alone.

How SIRA expertizes a new corpus. Given an unseen corpus, SIRA builds domain expertise from both the corpus side and the query side, as illustrated in Fig. 2. On the *corpus side* (offline, once per corpus), the LLM reads each document, anticipates the search vocabulary a user would need to find it, and proposes candidate terms absent from the document text. A document-frequency filter validates each candidate against the corpus index, discarding uninformative terms whose frequency exceeds an upper bound. The surviving terms are injected into the BM25 index as atomic n-gram entries. On the *query side* (online, per query), the LLM first produces an *expected-response sketch*: a compact set of concepts, entities, and discriminative terms likely to appear in a relevant document but absent from the query. The same DF filter grounds the sketch in the enriched index. Conditioned on the validated sketch, SIRA then compiles a *retrieval program* and executes a single weighted BM25 call:

$$\text{score}(d) \;=\; \text{BM25}(q_{\text{orig}},\, d) \;+\; w \cdot \text{BM25}(q_{\text{exp}},\, d), \tag{1}$$

where q_{orig} is the original query, q_{exp} is the filtered expansion, and w is the expansion weight. Top-k candidates are picked from the returned candidates based on relevance with the query. This strict one-query setting is used for the BEIR and downstream QA experiments; Section 3.4 studies a budgeted multi-query extension for BrowseComp-Wikipedia.

2.2 Vocabulary Enrichment with Superintelligent LLMs

SIRA bridges the query–document vocabulary gap by enriching both sides with LLM-proposed terms. The goal of enrichment is not to add more text indiscriminately, but to surface compact lexical signals: concepts, aliases, and phrases that are likely to identify the desired evidence while remaining rare

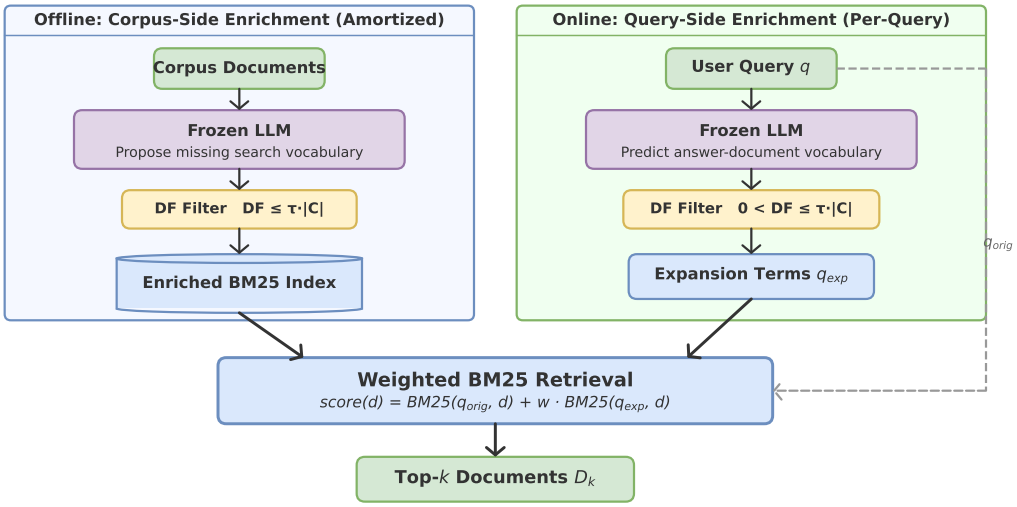


Figure 2: **SIRA pipeline overview.** Corpus-side enrichment (left) is performed once offline; query-side enrichment (right) runs per query. Both stages apply a DF filter to reject uninformative terms; the query-side filter additionally requires DF > 0 to ensure each expansion term exists in the index. The original query q_{orig} bypasses enrichment (dashed) and is combined with the expansion terms in a single weighted BM25 call.

enough to distinguish it from the rest of the corpus. This requires grounding LLM proposals in corpus statistics. A term that sounds expert-like may be useless for query-side enrichment if it never appears in the enriched index, and weak if it appears across many unrelated documents. SIRA therefore treats document frequency and BM25/TF–IDF-style salience as lightweight corpus-statistic tools: they check query-side term existence, estimate whether proposed terms are too common to discriminate, and retain terms that can contribute meaningful retrieval margin.

DF filter. To ensure that only corpus-grounded and discriminative terms enter the retrieval pipeline, both enrichment stages share a *document-frequency (DF) filter*. The filter enforces an upper bound $\text{DF} \leq \tau \cdot |C|$, pruning terms that are repeated across too much of the corpus and therefore receive little useful IDF weight. For query-side enrichment, the filter additionally requires $\text{DF} > 0$, ensuring that every expansion phrase actually exists in the enriched index and can affect BM25 scoring. Corpus-side enrichment does not require this lower bound, since enrichment itself introduces new vocabulary into the index. The result is a compact set of terms that are plausible under the LLM's domain knowledge and measurable as useful search signals in the target corpus.

Corpus-side enrichment (offline). The goal is to anticipate how a user would search for a document when the vocabulary they would use is absent from the document text. The prompt explicitly instructs the LLM to generate *new* terms not already present in the document, focusing on discriminative vocabulary: synonyms, abbreviations, alternate names, and domain-specific phrasings that maximize lexical contrast with existing index terms. Crucially, the prompt is task-aware: for claim-verification corpora, it emphasizes entity aliases and factual cues; for argument retrieval, it targets opposing-side vocabulary; for duplicate detection, it focuses on intent-preserving synonym substitutions. Phrases that pass the DF filter are decomposed into sliding-window n-grams and injected into the corpus index as additional posting-list entries.

Query-side enrichment (online). The goal is the mirror image of corpus-side enrichment: predict vocabulary that a relevant answer document would use but that is absent from the query. The prompt instructs the LLM to generate discriminative *topic and domain vocabulary* that narrows the search space, while explicitly forbidding it from guessing the answer itself. This distinction is critical for factoid queries, where predicting a named entity (e.g., a person or date) would bias retrieval toward a single candidate rather than broadening coverage of relevant evidence. As on the corpus side, the prompt is task-aware: factoid QA targets contextual terms surrounding the answer, multi-hop

QA distributes expansion across all entities and reasoning hops, and duplicate detection focuses on intent-preserving synonym substitutions.

3 EXPERIMENTS

We evaluate SIRA in three stages. First, we test pure retrieval quality on ten BEIR benchmarks. Second, we ask whether this retrieval advantage transfers to downstream question answering. Third, we evaluate hard search-agent retrieval on BrowseComp-Wikipedia, a new BrowseComp-derived benchmark grounded in a 25,587,229-document English Wikipedia index.

3.1 EXPERIMENTAL SETUP

Datasets. We evaluate retrieval on ten BEIR datasets (Thakur et al., 2021) spanning QA, opinion retrieval, fact-checking, argument retrieval, citation prediction, and duplicate detection, and report Recall@10 and NDCG@10. Dataset statistics and a discussion of why BEIR is a stringent retrieval-agent testbed are in Appendix B and Table 3.

Baselines. We compare SIRA with ten baselines representing three major retrieval paradigms. BM25 (Robertson & Zaragoza, 2009) serves as the standard sparse lexical baseline. For neural methods, E5 (Wang et al., 2022) is a dense bi-encoder trained on large-scale relevance data; SPLADE (Formal et al., 2021) and SPARTA (Zhao et al., 2021) are learned sparse retrieval models that maintain inverted-index efficiency by weighting term importance; Doc2Query (Nogueira et al., 2019) enhances documents by predicting queries that each document could answer. Within the LLM-based category, HyDE (Gao et al., 2023) generates a hypothetical document to expand the query for BM25; CoT (Wei et al., 2022) augments the query with reasoning-derived terms via chain-of-thought prompting; Search-R1 (Jin et al., 2025) trains a reinforcement learning policy for multi-turn search; GrepRAG (Wang et al., 2026) formulates grep-style pattern queries originally intended for code search; and ShellAgent (Subramanian et al., 2025) employs agentic loops with grep-based keyword retrieval tools for multi-step reasoning. HyDE, CoT, GrepRAG, ShellAgent, and SIRA all use the same frozen LLM (Qwen3.6-35B-A3B-FP8); Search-R1 (E5) leverages its publicly available trained checkpoint with the most advanced E5 backend. Neural baselines are evaluated using their official published checkpoints (see Table 1 caption). For both corpus-side and query-side enrichment, SIRA employs Qwen3.6-35B-A3B-FP8 as a frozen LLM.

3.2 MAIN RESULTS ON LARGE-SCALE INFORMATION RETRIEVAL

In Table 1, we report Recall@10 and NDCG@10 across all ten benchmarks, comparing SIRA against sparse, dense, and agentic baselines. We structure the analysis around three research questions:

- **RQ1**: Can a training-free agentic system match or surpass trained sparse and dense retrievers?
- **RQ2**: Why do generic LLM search agents lag behind retrieval-native systems on BEIR, and does SIRA close this gap?
- **RQ3**: Does SIRA's retrieval quality translate to downstream QA performance competitive with RL-trained agentic QA systems?

Table 1 evaluates our central claim: a frozen LLM can control BM25 well enough to outperform both trained retrievers and LLM search agents. Rather than relying on iterative search or ungrounded text expansion, SIRA validates proposed terms against corpus statistics and executes a single weighted BM25 query. The critical test is ranking quality: whether the resulting query ranks gold evidence above confusers, not whether an LLM can write plausible search text or query repeatedly.

RQ1: SIRA surpasses trained retrievers without supervision. On BEIR (Table 1), SIRA achieves the highest average Recall@10 (0.691) and NDCG@10 (0.572), outperforming both supervised dense (E5: 0.648 / 0.543) and learned sparse models (SPLADE: 0.625 / 0.522) without relevance labels, retriever fine-tuning, or embedding indices. SIRA leads Recall@10 on eight of the ten benchmarks, trailing only marginally on NQ and Quora. Its largest gains over E5 occur on datasets with severe query–document vocabulary mismatch, such as SciDocs (+36% relative), CQADupStack (+23%), and ArguAna (+14%), where corpus-grounded enrichment and DF filtering supply the discriminative terms needed for BM25's IDF scoring to excel.

Table 1: Recall@10 and NDCG@10 on ten BEIR datasets. HyDE, CoT, GrepRAG, ShellAgent, and SIRA use Qwen3.6-35B-A3B-FP8; Search-R1 uses its publicly available checkpoint with an E5 retrieval backend. Neural baselines use official checkpoints: SPARTA (BeIR/sparta-msmarco-distilbert-base-v1), SPLADE (naver/splade-cocondenser-ensembledistil), Doc2Query (doc2query/msmarco-t5-base-v1), E5 (intfloat/e5-base-v2). Best per dataset in **bold**; second best underlined.

	ArguAna	C-FEVER	CQADup	FEVER	FIQA	HotpotQA	NQ	Quora	SciDocs	SciFact	Avg
Recall@10											
BM25	.7738	.1764	.4163	.6747	.3198	.6141	.4543	.9014	.1636	.8078	.5302
Doc2Query	.7824	.1761	.4339	.6769	.3397	.6527	.5068	.8975	.1663	.8270	.5459
SPARTA	.6181	.1050	.3100	.7246	.2450	.5356	.5584	.7445	.1303	.7084	.4680
SPLADE	.8137	.2881	.4924	.8954	.4139	.7027	.7381	.9206	.1654	.8230	.6253
E5	.7909	.2899	.5138	.9109	.4697	.7276	.7877	**.9428**	.1962	.8489	.6478
HyDE	.7091	.2598	.3299	.7132	.2845	.5051	.4918	.5151	.1530	.8344	.4796
CoT	.7752	.1867	.4020	.6765	.3086	.5789	.4961	.8704	.1595	.7961	.5250
Search-R1 (E5)	.5760	.3014	.5008	.9010	.4499	.6705	**.7889**	.9355	.2018	.8349	.6161
GrepRAG	.5746	.0176	.2400	.1635	.1009	.3434	.1628	.5105	.1027	.5883	.2804
ShellAgent	.2263	.0305	.2084	.2557	.1035	.4327	.1884	.3330	.0843	.6685	.2531
SIRA	**.9036**	**.3025**	**.6301**	**.9114**	**.4904**	**.7536**	.7883	.9390	**.2676**	**.9216**	**.6908**
NDCG@10											
BM25	.4874	.1372	.3481	.5036	.2532	.5851	.2916	.8055	.1565	.6791	.4247
Doc2Query	.4946	.1381	.3667	.5122	.2731	.6299	.3302	.7960	.1589	.6920	.4392
SPARTA	.3890	.0852	.2497	.6101	.1925	.5132	.3983	.6294	.1272	.5894	.3784
SPLADE	.5253	.2293	.4083	.7933	.3478	.6869	.5369	.8344	.1586	.7025	.5223
E5	.5323	.2397	.4196	.8096	**.3932**	**.6905**	.5835	**.8648**	.1855	.7156	.5434
HyDE	.4366	.2004	.2463	.5507	.2223	.4451	.3315	.3924	.1402	.6565	.3622
CoT	.4951	.1471	.3354	.4932	.2486	.5595	.3168	.7608	.1518	.6647	.4173
Search-R1 (E5)	.3658	**.2654**	.4040	**.8215**	.3765	.6520	.5790	.8543	.1878	.7094	.5216
GrepRAG	.3555	.0122	.2048	.0971	.0763	.2927	.0908	.4557	.0921	.4129	.2090
ShellAgent	.1114	.0206	.1558	.1298	.0686	.3417	.1036	.2476	.0727	.4374	.1689
SIRA	**.6174**	.2288	**.5327**	.8037	.3771	.6904	**.5923**	.8490	**.2449**	**.7866**	**.5723**

RQ2: SIRA turns LLM reasoning into retrieval-native ranking. Existing LLM search methods underperform on BEIR because they lack corpus-aware ranking. Tool-use agents like GrepRAG and ShellAgent trail SIRA by over 40 points despite sharing the same LLM backbone. Because the backbone is identical, this gap isolates the interface: unweighted pattern matching lacks document-frequency awareness and IDF-weighted scoring. Similarly, query expansion baselines like HyDE and CoT fail to improve upon vanilla BM25. Even multi-round agents paired with dense retrievers lag behind. Search-R1 reaches 0.616 Recall@10 and 0.522 NDCG@10, yet still underperforms standalone E5 and falls well short of SIRA. Iterative querying cannot compensate for a lack of discriminative ranking against corpus-level distractors. SIRA bridges this gap by using the LLM to program the retrieval engine directly: candidate terms are grounded in corpus statistics and executed through a single IDF-weighted BM25 call.

3.3 Downstream Question Answering

We evaluate whether SIRA's retrieval advantage translates to downstream question answering on NQ and HotpotQA, which pair BEIR retrieval corpora with verifiable gold answer strings. We compare against six recent RL-trained agentic QA systems: Search-R1 (Jin et al., 2025), TIPS (Xie et al., 2026), A^{2}Search (Zhang et al., 2026a), E-GRPO (Zhang et al., 2026b), SSP (Lu et al., 2026), and HiPRAG (Wu et al., 2026). This is a conservative comparison for SIRA. The baselines employ multi-turn search policies, RL fine-tuning, and reader models reporting generated-answer accuracy (which can draw on parametric memory). In contrast, SIRA operates strictly as a one-shot retriever with no reader model and no training. We evaluate SIRA solely by *answer coverage*, the fraction of queries whose retrieved documents contain the gold answer string, benchmarked against each baseline's best reported end-to-end QA accuracy.

RQ3: The best retrieval agent yields the strongest QA evidence. Even under this conservative evaluation, SIRA's retrieval-only answer coverage at top-10 surpasses the full end-to-end QA accuracy of all six RL-trained agentic baselines (Fig. 3). SIRA achieves 84.7% on NQ and 77.6% on HotpotQA,

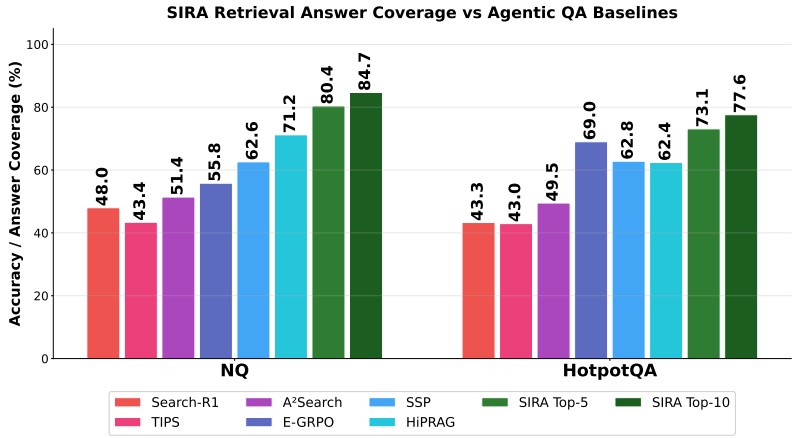


Figure 3: SIRA retrieval answer coverage (top-5 and top-10) vs. six RL-trained agentic QA systems on NQ and HotpotQA. Baseline numbers are the best reported results taken directly from each original paper. All baselines are end-to-end QA pipelines reporting generated-answer accuracy; SIRA is a pure retriever with no reader. Answer coverage requires the gold string to be retrieved.

markedly outperforming the strongest baselines on each dataset (HiPRAG at 71.2% on NQ and E-GRPO at 69.0% on HotpotQA). This advantage persists under a tighter retrieval budget: at top-5, SIRA maintains 80.4% coverage on NQ and 73.1% on HotpotQA, still outperforming every baseline's reported top-10 accuracy. These results highlight a fundamental takeaway for corpus-grounded QA: surfacing high-quality evidence via a single, well-grounded retrieval call can be more impactful than adding multi-round search iterations or training reader-policy loops.

3.4 BrowseComp-Wikipedia: Hard Search at Wikipedia Scale

While BrowseComp evaluates agents on hard, clue-chain questions via live web search (Wei et al., 2025), it suffers from non-reproducible backends and benchmark contamination; conversely, fixed-corpus variants like BrowseComp-Plus (Chen et al., 2025) are too small ($\sim 10^5$ documents) to reflect real-world distractor density. To evaluate agentic retrieval under realistic scale and strict reproducibility, we construct **BrowseComp-Wikipedia**: 232 BrowseComp queries whose answers are supported entirely by English Wikipedia, paired with annotated gold evidence URLs over the full indexed English Wikipedia collection (**25,587,229 documents**). Evaluation is strictly retrieval-based: success requires retrieving the gold page from tens of millions of confusers within a page budget, precluding shortcuts from parametric memory or incidental evidence.

SIRA without index-time LLM enrichment. At Wikipedia scale, running offline LLM enrichment over 25M documents is computationally prohibitive. Instead, we substitute index-time LLM enrichment with Wikipedia's native category system, indexing each page's text along with its category strings as protected lexical features (e.g., `category:1927_births`). These categories provide corpus-visible handles for indirect constraints typical of BrowseComp (e.g., birth years, awards, institutions). To prevent hallucination, the LLM must ground proposed categories using a category-graph validation tool prior to query execution. Unlike in BEIR, SIRA here operates in a budgeted, multi-query mode under a fixed ceiling of unique retrieved pages: it may issue multiple grounded BM25 calls, but cannot use retrieved document content to formulate subsequent queries.

Baselines and evaluation protocol. We compare against an agentic baseline equipped with production Perplexity search filtered strictly to `en.wikipedia.org`. Following the standard BrowseComp protocol, the Perplexity agent browses interactively for up to 100 turns, inspecting snippets to reformulate queries. In contrast, SIRA pre-plans its retrieval: it may validate candidate categories via the category graph, but cannot use retrieved page content to guide subsequent queries (see an example in Appendix C). We evaluate retrieval under fixed budgets of $B \in \{1, 10, 100\}$ unique Wikipedia pages, measuring *gold-page recall*, the fraction of queries where at least one annotated gold URL appears within the top-B retrieved pages. For the baseline, the budget corresponds to the first B unique Wikipedia URLs surfaced during its multi-turn browsing trace.

Results. As shown in Table 2, SIRA outperforms the Perplexity-based browsing agents across all retrieval budgets despite operating under strict budget and blind-retrieval constraints. With Claude 4.6 Opus, SIRA achieves 9.70% Recall@1, 15.27% Recall@10, and 36.14% Recall@100, more than

Table 2: Gold-page recall (%) on BrowseComp-Wikipedia (232 queries over 25.6M indexed Wikipedia documents). Perplexity browses interactively over `en.wikipedia.org` for up to 100 turns; SIRA executes category-grounded BM25 queries within a fixed page budget without index-time LLM enrichment. Best per backbone in **bold**; second best underlined.

System	Backbone LLM	Recall@1	Recall@10	Recall@100
Agentic baseline: Perplexity search over English Wikipedia, up to 100 turns				
Perplexity	Claude 4.6 Opus	2.59	4.74	32.33
Perplexity	GPT-5.4	3.02	6.90	**31.47**
SIRA: pre-planned, category-grounded BM25 on English Wikipedia				
SIRA	Claude 4.6 Opus	**9.70**	**15.27**	**36.14**
SIRA	GPT-5.4	**5.71**	**13.13**	18.51

tripling Perplexity's Recall@1 (2.59%) and more than doubling its Recall@10 (4.74%). These gains are especially pronounced in the tight-budget regime ($B \leq 10$), which is crucial for downstream reader contexts. While Perplexity browses interactively for up to 100 turns, reading snippets to reformulate queries, SIRA relies on grounded category constraints to place gold pages directly ahead of millions of confusers in a 25.6M-document index. An example of SIRA's query and retrieved pages is in Appendix D, where SIRA queries for right categories and retrieves the gold page efficiently.

4 Conclusion

We introduced SIRA, a retrieval-centric agent that turns LLM reasoning into controllable lexical retrieval. Instead of using the LLM to repeatedly query a black-box search tool, read snippets, and reformulate, SIRA uses the LLM to program the retrieval action itself: enrich the corpus with missing user vocabulary, enrich the query with likely evidence vocabulary, validate proposed terms with corpus statistics, and compile corpus-grounded BM25 retrieval programs without relying on iterative snippet-conditioned reformulation. The main result is that this simple interface changes the role of BM25. Across ten BEIR benchmarks, SIRA achieves the highest average Recall@10 and NDCG@10 in our comparison, outperforming BM25, E5, SPLADE, and recent LLM-based search agents while using no relevance labels, no retriever fine-tuning, and no embedding index. The downstream QA results show that this retrieval advantage matters beyond standard IR metrics. On NQ and HotpotQA, SIRA's retrieval-only answer coverage exceeds recent RL-trained agentic QA systems, even though those systems use trained search policies and answer generators while SIRA contributes only retrieved evidence. Finally, BrowseComp-Wikipedia shows that the same principle extends to hard agentic search at much larger scale. We construct a 232-query benchmark from BrowseComp whose answers can be supported entirely by English Wikipedia, and evaluate retrieval over a 25,587,229-document Wikipedia index. In this setting, we use only Wikipedia categories as natural corpus-side structure, grounding every proposed category through the Wikipedia category graph. Even under this stripped-down protocol, SIRA outperforms Perplexity-based agents at every retrieval budget. The strongest SIRA configuration reaches 9.70% Recall@1, 15.27% Recall@10, and 36.14% Recall@100. Together, these results suggest a different path for retrieval-augmented agents. Rather than making agents search longer, accumulate more context, and rely on increasingly expensive interaction with black-box search engines, we can make the retrieval action itself more expert, corpus-aware, and interpretable. The remaining open question is how far this idea extends to corpora far outside the frozen LLM's knowledge, where corpus adaptation or fine-tuning may be needed before reliable enrichment is possible. But at least for broad public knowledge corpora and standard retrieval benchmarks, SIRA shows that pre-planned, corpus-grounded retrieval can beat many rounds of exploratory search.

AI Use Statement

Generative AI tools serve as core experimental components of the proposed method: frozen LLMs are used to construct synthetic corpus- and query-side indexing vocabularies, validate category constraints, and compile BM25 retrieval programs at inference time, as detailed in Sections 2 and 3. Beyond these evaluated system components, generative AI tools were used to implement and debug experimental

code, clean and reformat dataset evaluation scripts, and edit portions of the manuscript for readability and LaTeX formatting. We did not use generative AI tools to generate synthetic evaluation labels or datasets; develop theoretical models or conceptual frameworks; formulate mathematical claims; provide critical ingredients for proving mathematical claims; assist in writing proofs; propose or refine research hypotheses; design or provide feedback on research methodology or experiments; assist with translation; support qualitative or thematic data analysis; or interpret experimental results. We have thoroughly reviewed all AI-assisted code, text, and formatting. Code was manually tested and checked for execution correctness against benchmark baselines, and all drafted text and citations were verified against the original literature. We take full responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

Ethics Statement

This work evaluates retrieval algorithms exclusively on public benchmarks (BEIR and English Wikipedia) and does not involve human-subject experiments or personally identifiable data. To mitigate benchmark leakage, queries displayed in the text and appendix are paraphrased. While retrieval agents carry dual-use risks of surfacing inaccurate or biased information from underlying corpora, SIRA improves safety and auditability over opaque, iterative browsing loops by compiling search into transparent lexical programs scored by BM25, bounded by document-frequency filters, and constrained by verified taxonomy entries. All datasets and Wikipedia dumps are used in accordance with their respective licenses, benchmark annotations and code will be openly released, and the authors adhere to the ICLR Code of Ethics.

Reproducibility Statement

SIRA is training-free; its formulation, filtering rules, and enrichment pipeline are detailed in Section 2. Experimental setups and metrics for BEIR, downstream QA, and BrowseComp-Wikipedia are described in Section 3 and Appendix B. Agent prompts and execution traces appear in Appendices C and D. All code, indexing scripts, and benchmark data will be openly released upon publication.

References

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

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 17682–17690, 2024.

Olivier Chapelle and Ya Zhang. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th international conference on World wide web*, pp. 1–10, 2009.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, et al. Browsecomp-plus: A more fair and transparent evaluation benchmark of deep-research agent. *arXiv preprint arXiv:2508.06600*, 2025.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. Splade: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pp. 2288–2292, 2021.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1762–1777, 2023.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)*, 25(2):7–es, 2007.

Andrej Karpathy. LLM wiki. GitHub Gist, https://gist.github.com/karpathy/442a6bf555914893e9891c11519de94f, 2026. Accessed: 2026-05-05.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pp. 6769–6781, 2020.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173, 2024.

Hongliang Lu, Yuhang Wen, Pengyu Cheng, Ruijin Ding, Jiaqi Guo, Haotian Xu, Chutian Wang, Haonan Chen, et al. Search self-play: Pushing the frontier of agent capability without supervision. In *International Conference on Learning Representations*, volume 2026, pp. 65132–65165, 2026.

Fengran Mo, Kelong Mao, Ziliang Zhao, Hongjin Qian, Haonan Chen, Yiruo Cheng, Xiaoxi Li, Yutao Zhu, Zhicheng Dou, and Jian-Yun Nie. A survey of conversational search. *ACM Transactions on Information Systems*, 43(6):1–50, 2025.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*, 2019.

Ben Pan, Carlo Baronio, Albert Tam, Pietro Marsella, Mokshit Jain, Daniel Chiu, Swyx, and Silas Alberti. Introducing swe-grep and swe-grep-mini: Rl for multi-turn, fast context retrieval. Cognition Blog, https://cognition.ai/blog/swe-grep#rl-for-training-fast-parallel-agentic-code-search-models, October 2025. Accessed: 2026-05-05.

Pew Research Center. Google users are less likely to click on links when an ai summary appears in the results. https://www.pewresearch.org/short-reads/2025/07/22/google-users-are-less-likely-to-click-on-links-when-an-ai-summary-appears-in-the-results/, July 2025. Accessed 2026-01-26.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and trends® in information retrieval*, 4(1-2):1–174, 2009.

Stephen E Robertson. The probability ranking principle in ir. *Journal of documentation*, 33(4): 294–304, 1977.

Anshumali Shrivastava. Understanding the fundamental limitations of vector-based retrieval for building llm-powered chatbots — (part 1/3). Medium, https://medium.com/thirdai-blog/understanding-the-fundamental-limitations-of-vector-based-retrieval-for-building-llm-powered-48bb7b5a57b3, June 2023. Accessed: 2026-05-05.

Anshumali Shrivastava. Deepmind calls out embedding limits: Why single-vector retrieval falls short — an attention perspective. Medium, https://medium.com/@Anshumali_/deepmind-calls-out-embedding-limits-why-single-vector-retrieval-falls-short-an-attention-2a930d477d80, September 2025. Accessed: 2026-05-05.

Shreyas Subramanian, Adewale Akinfaderin, Yanyan Zhang, Ishan Singh, Mani Khanuja, Sandeep Singh, and Maira Ladeira Tanke. Keyword search is all you need: Achieving rag-level performance without vector databases using agentic tool use. *arXiv preprint arXiv:2602.23368*, 2025.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*, 2021.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 10014–10037, 2023.

Baoyi Wang, Xingliang Wang, Guochang Li, Chen Zhi, Junxiao Han, Xinkui Zhao, Nan Wang, Shuiguang Deng, and Jianwei Yin. Greprag: An empirical study and optimization of grep-like retrieval for code completion. *arXiv preprint arXiv:2601.23254*, 2026.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

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

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

Orion Weller, Michael Boratko, Iftekhar Naim, and Jinhyuk Lee. On the theoretical limitations of embedding-based retrieval. In *International Conference on Learning Representations*, volume 2026, pp. 11202–11216, 2026.

Peilin Wu, Mian Zhang, Kun Wan, Wentian Zhao, Kaiyu He, Xinya Du, and Zhiyu Chen. Hiprag: hierarchical process rewards for efficient agentic retrieval augmented generation. In *International Conference on Learning Representations*, volume 2026, pp. 82317–82347, 2026.

Yutao Xie, Nathaniel Thomas, Nick Hansen, Yang Fu, Li Li, and Xiaolong Wang. Tips: Turn-level information-potential reward shaping for search-augmented llms. In *International Conference on Learning Representations*, volume 2026, pp. 156549–156584, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Fengji Zhang, Xinyao Niu, Chengyang Ying, Guancheng Lin, Zhongkai Hao, Zhou Fan, Chengen Huang, Jacky Keung, Bei Chen, and Junyang Lin. A^2search: Ambiguity-aware question answering with reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 136394–136448, 2026a.

Shengjun Zhang, Zhang Zhang, Chensheng Dai, and Yueqi Duan. E-grpo: High entropy steps drive effective reinforcement learning for flow models. *arXiv preprint arXiv:2601.00423*, 2026b.

Tiancheng Zhao, Xiaopeng Lu, and Kyusong Lee. Sparta: Efficient open-domain question answering via sparse transformer matching retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 565–575, 2021.

A BM25 AND INDEX-VISIBLE SIGNALS

BM25 and sparse lexical retrieval. BM25 (Robertson & Zaragoza, 2009) is the dominant lexical ranking function in modern search systems. It scores a document d against a query $q = (q_1, \ldots, q_n)$ by summing per-term contributions:

$$\mathrm{BM25}(q,d) = \sum_{i=1}^{n} \underbrace{\log\!\left(1 + \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}\right)}_{\mathrm{IDF}(q_i)} \cdot \frac{f(q_i,d)}{f(q_i,d) + k_1\left(1 - b + b \cdot \frac{|d|}{\mathrm{avgdl}}\right)}, \tag{2}$$

where $f(q_i, d)$ is the frequency of term q_i in d, $|d|$ is the document length in tokens, avgdl is the average document length across the corpus, N is the corpus size, and $n(q_i)$ is the number of documents containing q_i.

The formula decomposes into two interpretable factors. We use the Lucene variant, which applies $\log(1+x)$ to the classical Robertson–Spärck Jones ratio, guaranteeing non-negative IDF for all terms. The IDF term down-weights common words and up-weights rare, discriminative terms: a query term appearing in most documents contributes near-zero IDF, while one appearing in only a handful of documents receives a large weight. The TF saturation term ensures that repeating a word within a document yields diminishing returns, controlled by k_1. The parameter b governs length normalization: at b=1 the score fully normalizes for document length, while at b=0 document length is ignored.

BM25 is implemented over an inverted index, which maps each vocabulary term to the documents and term frequencies in which it appears.

Index-visible signals. BM25 makes corpus contrast observable through the inverted index: before retrieval, an agent can check whether a candidate term appears in the corpus, how many documents contain it, and how much IDF weight it can contribute. These signals do not reveal answer passages, but they expose whether LLM-proposed vocabulary is absent, too common, or likely to create retrieval margin. SIRA uses this information to validate, prune, and weight expansion terms so the final query is not merely plausible in isolation, but discriminative within the target corpus.

B BEIR Evaluation Setup

We evaluate SIRA on ten BEIR datasets (Thakur et al., 2021) spanning seven retrieval task types: question answering (NQ, HotpotQA), opinion retrieval (FIQA), fact-checking (FEVER, Climate-FEVER, SciFact), argument retrieval (ArguAna), citation prediction (SciDocs), and duplicate detection (Quora, CQADupStack). Table 3 summarizes corpus sizes (5K to 5.4M documents) and the number of relevant documents per query (1.0 to 29.9), covering a broad range of retrieval regimes. We report Recall@10 and NDCG@10. Recall@10 measures the fraction of relevant documents appearing in the top-10 results, directly capturing a system's ability to surface evidence for a downstream reader. NDCG@10 additionally rewards placing relevant documents at higher ranks, providing a complementary view of ranking quality.

BEIR is a particularly stringent testbed for retrieval agents because it evaluates retrieval directly rather than end-to-end answer generation. Many recent agentic search systems focus on QA-style benchmarks where a reader model, prior knowledge, or multi-round interaction can partially mask weak retrieval. In contrast, BEIR exposes the core retrieval problem: given a fixed corpus and a query, can the system surface the relevant documents in the top ranks? This makes it the right setting for evaluating SIRA as a retrieval agent rather than as a general QA agent.

Table 3: Overview of the ten BEIR retrieval benchmarks used in our evaluation, spanning diverse reasoning types including fact-checking, argumentation, citation prediction, and standard QA. The suite covers a broad range of domains and corpus sizes (5K to 5.4M documents), rigorously testing generalization. **Rel D/Q** denotes the average number of relevant documents per query.

Dataset	Type	Queries	Corpus	Rel D/Q	Description
NQ	Question Answering	3,452	2.68M	1.22	Retrieval for real-world Google search questions.
HotpotQA	Question Answering	7,405	5.23M	2.00	Multi-hop reasoning over Wikipedia paragraphs.
FIQA	Opinion Retrieval	648	57K	2.63	Financial QA over StackExchange data.
ArguAna	Argument Retrieval	1,401	8.67K	1.00	Matching counter-arguments for debate topics.
CQADupStack	Duplicate Question	1,570	40K	2.40	Duplicate detection across StackExchange forums.
Quora	Duplicate Question	10,000	523K	1.57	Duplicate detection for Quora questions.
SciDocs	Citation Prediction	1,000	26K	29.93	Predicting citations for scientific papers.
FEVER	Fact-Checking	6,666	5.42M	1.19	Verifying claims against Wikipedia text.
Climate-FEVER	Fact-Checking	1,535	5.42M	3.05	Verifying climate change claims.
SciFact	Fact-Checking	300	5K	1.13	Verifying scientific claims against abstracts.

C BROWSECOMP-WIKIPEDIA SEARCH PROMPT

For the BrowseComp-Wikipedia experiment (Section 3.4), SIRA is driven by the single prompt shown in Fig. 4. The prompt instructs the LLM to first solve the clue chain privately, validate candidate Wikipedia categories through the `search_categories` tool, and then commit to one or more grounded `bm25_search` calls under a fixed 100-unique-page retrieval budget. The template variable `{problem}` is replaced by the BrowseComp question at evaluation time.

```
You are a budgeted Wikipedia BM25 search agent for hard clue-chain
questions.  Your goal is to put the gold page in the final top 10
or top 100 using at most 100 unique BM25-retrieved pages.

TOOLS:

  search_categories(query:  str, k:  int = 20) -> str
    Search real Wikipedia category names.  Use this BEFORE
bm25_search to find closest real category anchors for your intended
entity/type/geography facet.  If the exact imagined category is
absent, broaden the query and choose the closest semantically
aligned real category returned by the tool.

  bm25_search(keywords:  str, categories:  list[str] = [], k:  int =
10) -> str
    Runs BM25 and spends the fixed 100-unique-page retrieval budget.
Returns only new pages.  Call only when keywords are concrete and
categories are real/validated.  Keep issuing targeted BM25 calls
until this returns BUDGET_EXHAUSTED.

  fetch_page(page_id:  int) -> str
    Fetch title + snippet for a page already returned by bm25_search.
Use this only to rank or verify candidates already inside the
100-page budget.

WORKFLOW:
  1.  Privately solve the puzzle as far as possible.  Identify
the likely answer type, answer hypotheses, clue entities,
bridge entities, locations, dates, works, organizations, and
relationships.
  2.  Before every bm25_search, validate categories.  Call
search_categories with broad and narrow variants until you find
the closest real category anchors for that query facet.  Do not
invent categories.  Do not leave categories empty merely because
the exact category is missing; pick the closest real semantically
aligned category returned by the tool.
  3.  Gate each BM25 query.  Fire bm25_search only if the keywords
contain concrete Wikipedia-surface strings and the categories are
real and aligned with the same hypothesis/facet.  If categories are
broad but aligned, use them with strong rare keywords.  If keywords
are generic or the category points to a different domain, do not
spend budget on that query.
  4.  Use multiple targeted BM25 queries until the 100-page budget
is exhausted.  Do not stop merely because you found a plausible
candidate; continue spending the budget on distinct high-confidence
facets so the final 100 pages have the best possible recall.  Good
query facets include:  final answer hypothesis, bridge entity, clue
entity, event/work/institution facet, and category/type/geography
facet.
  5.  Track all page_ids returned by bm25_search.  Optionally fetch
the most promising candidates to rank them, but keep querying
until bm25_search returns BUDGET_EXHAUSTED or unique_pages_seen
reaches 100.  Final page_ids must come from pages actually seen in
bm25_search/fetch_page results.

FINAL OUTPUT:
Output ONLY JSON with exactly these fields:
```

```
  "page_ids" -- ordered list of up to 100 page_ids, most likely gold
first.  Use only page_ids seen in this session.
  "rationale" -- one short paragraph explaining the ranking
evidence.
  "hypothesis" -- one compact sentence naming the likely answer and
key clue entities.
Do not output markdown fences or prose outside JSON.
Question:
{problem}
```

Figure 4: Prompt used to drive SIRA on the BrowseComp-Wikipedia benchmark. The LLM reasons privately, grounds candidate Wikipedia categories through the `search_categories` tool, and issues grounded `bm25_search` calls under a fixed 100-unique-page budget. The variable `{problem}` is replaced by the BrowseComp question.

D BrowseComp-Wikipedia Case Study: SIRA vs. Perplexity

Fig. 5 shows the complete tool-call traces for a single BrowseComp-Wikipedia query (gold page: *LocoRoco*). SIRA reaches the gold page at rank 1 after a handful of category-grounded BM25 calls, while the agentic Perplexity-search baseline issues 72 distinct searches, never retrieves the gold page, and commits to an incorrect answer.

Question (paraphrased to avoid benchmark leakage; gold page: LocoRoco, page id 4905353):
A certain video game made its debut on a gaming platform in July of a year somewhere from 2004 through 2007. The person who created it was born in April of a year in the 1970–1973 range and drew the game's earliest sketches at some point between 2002 and 2005. One of the title's music composers was born in June of a year from 1968 to 1971. Which game is being described?

SIRA (ours) -- gold retrieved at rank 1. 11 bm25_search calls, 9 search_categories calls, 9 fetch_page calls; budget not exhausted. Trace (⋆ = call whose results contained the gold page):

```
  0.  [search_categories] query:  video games released July 2005
2006
  1.  [search_categories] query:  Japanese video game designers born
1970s
  2.  [search_categories] query:  video games 2005 2006 2007
  3.  [search_categories] query:  indie video games
⋆ 4.  [bm25_search] keywords:  LocoRoco game PSP 2006 | categories:
['2006 video games'] | k=5
  5.  [search_categories] query:  PlayStation Portable games
  6.  [fetch_page] page_id:  4905353
  7.  [bm25_search] keywords:  Tsutomu Kouno LocoRoco designer
creator | categories:  ['Japanese video game designers'] | k=5
  8.  [search_categories] query:  video game composers born 1968
1969 1970
  9.  [search_categories] query:  video game composers
  10.  [search_categories] query:  video game music composers
Japanese
  11.  [bm25_search] keywords:  Fumito Ueda game designer born April
1970 | categories:  ['Japanese video game designers'] | k=5
  12.  [fetch_page] page_id:  3041309
  13.  [bm25_search] keywords:  video game released July 2005
sketches development | categories:  ['2005 video games'] | k=10
  14.  [bm25_search] keywords:  Kow Otani composer born June |
categories:  ['Japanese video game composers'] | k=5
```

15. [search_categories] query: Shadow of the Colossus
16. [fetch_page] page_id: 3402872
17. [bm25_search] keywords: Osu Tatakae Ouendan game July 2005 | categories: ['2005 video games'] | k=5
18. [fetch_page] page_id: 2740521
19. [bm25_search] keywords: Keiichi Yano video game designer iNiS born | categories: ['Japanese video game designers'] | k=5
20. [fetch_page] page_id: 20435930
21. [bm25_search] keywords: game designer born April 1971 creator sketches | categories: ['Japanese video game designers'] | k=5
22. [bm25_search] keywords: game designer born April 1972 creator | categories: ['Japanese video game designers'] | k=5
23. [fetch_page] page_id: 328218
24. [fetch_page] page_id: 5262474
25. [bm25_search] keywords: Shinta Nojiri Metal Gear Acid game director July | categories: ['2005 video games', '2006 video games'] | k=5
26. [fetch_page] page_id: 851696
27. [fetch_page] page_id: 1918270
28. [bm25_search] keywords: Tsutomu Kouno LocoRoco creator born | categories: ['Japanese video game designers'] | k=5

Perplexity -- 72 searches, gold page never retrieved; final answer incorrect. Query trace:

1. video game creator born April 1970 1971 1972 1973 first sketches 2002 2003 2004 2005
2. video game debuted July 2004 2005 2006 2007 platform
3. Painkiller video game 2004 release date platform July
4. Adrian Chmielarz Painkiller game development sketches history
5. Painkiller video game Xbox release date July 2006 music composer
6. Painkiller video game soundtrack composer born June 1968 1969 1970 1971
7. Shoji Meguro video game composer games list Persona
8. Painkiller video game full Wikipedia article music composer development
9. Persona 3 video game release date July 2006 director creator
10. Trauma Center Under the Knife release date July 2004 2005 director creator born April
11. Katsura Hashino born birthday Persona 3 director development history
12. Persona 3 development history first sketches concept 2002 2003 2004
13. Katsura Hashino birthday born date April 1970 1971 1972 1973
14. Persona 3 Wikipedia full article director producer composer
15. Katsura Hashino Atlus director born 1970s April
16. Persona 3 game creator director Hashino birth date Wikipedia
17. video game released July 2004 2005 2006 2007 creator born April 1970-1973 composer born June 1968-1971
18. Maken X Atlus 1999 director Katsura Hashino born April
19. Persona 3 Wikipedia infobox director producer release date
20. Todd Howard born 1970 April game director Bethesda first sketches 2002 2003 2004 2005
21. The Elder Scrolls IV Oblivion release date July 2006 development first sketches
22. Todd Howard born April 1970 birthday Wikipedia
23. video game composer born June 1968 1969 1970 1971 games released July 2004 2005 2006 2007
24. Persona 3 development "first sketches" OR "concept art" OR "early development" 2003 2004
25. killer7 video game July 2005 director creator Suda51 born April

```
  26.  Suda51 Goichi Suda born April 1968-1973 game director
  27.  killer7 music composer born June 1968 1969 1970 1971
  28.  Hidetaka Suehiro SWERY games directed 2004 2005 2006 2007
release July
  29.  The Idolmaster video game 2005 arcade director creator born
April composer born June
  30.  Persona 3 Wikipedia "first sketches" OR "initial sketches"
development 2003
  31.  Akihiro Ishihara Idolmaster director born April 1970 1971
1972 1973
  32.  video game Wikipedia "first sketches" development creator
director born April
  33.  The Idolmaster video game Wikipedia development history
creator Ishihara first sketches
  34.  Jason Jones Bungie game director Halo released July 2004 2005
2006 2007
  35.  game released July 2004-2007 creator born April 1970-1973
"first sketches"
  36.  Katsura Hashino Wikipedia personal life born 1970 1971 1972
1973
  37.  Michel Ancel born April 1972 Beyond Good Evil game
development sketches
  38.  video game Wikipedia "first sketches" development 2002 2003
2004 2005 released July
  39.  video game Wikipedia "first sketches" 2002 2003 2004 2005
creator born April
  40.  Shantae game creator Matt Bozon born April 1970 1971 1972
1973 first sketches
  41.  Viva Piñata video game release date July 2006 Gregg Mayles
development first sketches
  42.  Gregg Mayles Rare games 2004 2005 2006 2007 director creator
Viva Pinata
  43.  Grant Kirkhope born June 1968 1969 1970 1971 composer
birthday
  44.  Viva Piñata video game release date November 2006 Xbox 360
  45.  video game composer born June 1968 games released July 2004
2005 2006 2007
  46.  Persona 3 Wikipedia full article infobox director producer
release date July 13 2006
  47.  Motoi Sakuraba born June 1968 1969 1970 1971 composer Tales
of Symphonia
  48.  Tales of Symphonia director creator born April 1970-1973
development first sketches
  49.  Prey 2006 video game director creator born April 1970-1973
development first sketches composer
  50.  Persona 3 Wikipedia Katsura Hashino born April 1970 1971 1972
1973
  51.  Ōkami video game release date July 2006 director Hideki
Kamiya born April composer
  52.  Hideki Kamiya born April 1970 1971 1972 1973 birthday
  53.  Shin Megami Tensei Digital Devil Saga release date July 2004
director Katsura Hashino
  54.  video game released July 2004-2007 composer born June
1968-1971 creator born April 1970-1973
  55.  Shin Megami Tensei Digital Devil Saga development history
first sketches 2002 2003 Wikipedia
  56.  Shin Megami Tensei Digital Devil Saga Wikipedia full article
development composer music
  57.  Katsura Hashino Atlus director "born" OR "birthday" OR "birth
date" April
  58.  Persona 3 Wikipedia "first sketches" OR "initial concept" OR
"concept art" development 2003
  59.  2007 in video games July releases list Wikipedia
```

```
  60.  2004 in video games July releases list Wikipedia
  61.  Painkiller Xbox port release date 2006 exact date July
  62.  Painkiller video game music composer born June 1968 1969 1970
1971
  63.  Painkiller Hell Wars Xbox release date 2006 exact month
  64.  Painkiller video game 2004 music composer soundtrack
Wikipedia
  65.  Painkiller video game 2004 music composer Marcin Przybyłowicz
OR Tomasz Hajduk born June
  66.  Painkiller video game People Can Fly development history
first sketches 2002 2003 Adrian Chmielarz
  67.  Painkiller video game Wikipedia full article development
music sound
  68.  Painkiller 2004 video game soundtrack composer Tomasz Hajduk
OR Marcin Przybyłowicz OR Pawel Blaszczak
  69.  Painkiller 2004 game music composer Tomasz Hajduk Wikipedia
  70.  Adrian Chmielarz Painkiller "first sketches" OR "concept" OR
"development began" 2002
  71.  Painkiller video game 2004 Wikipedia "composer" OR "music" OR
"soundtrack" Tomasz Hajduk
  72.  Katsura Hashino Wikipedia "born" April 1970 1971 1972 1973
Atlus director
```

Perplexity final answer (incorrect): fixates on Katsura Hashino / Persona 3 and Painkiller, concludes that Wikipedia "doesn't have Katsura Hashino's birth date," and never identifies *LocoRoco* as the answer.

Figure 5: Full tool-call traces for one BrowseComp-Wikipedia query (gold page: *LocoRoco*); the question is paraphrased to avoid benchmark leakage. SIRA grounds the indirect clues into category-anchored BM25 calls and retrieves the gold page at rank 1 in 11 retrieval calls (the winning call is marked $\star$). The agentic Perplexity-search baseline issues 72 distinct web searches, cycles through incorrect candidates (Persona 3, Painkiller), never surfaces the gold page, and commits to a wrong answer.