SUPERINTELLIGENT RETRIEVAL AGENT: THE NEXT FRONTIER OF INFORMATION RETRIEVAL

Anonymous authors
Paper under double-blind review

ABSTRACT

Retrieval-augmented agents are increasingly the interface to large organizational knowledge bases, yet most treat retrieval as a black box: they can rewrite queries and react to returned snippets, but they cannot directly steer retrieval (e.g., enforce constraints, weight keywords, or decompose queries) or exploit what is already likely known in the domain. In practice, these agents act like a domain newcomer, relying on standard textbook commonsense resulting in inefficient exploratory querying; instead, our proposal behaves like an expert by first anticipating the likely answer and then using that expectation to plan and precisely control retrieval. We introduce SIRA (SuperIntelligent Retrieval Agent), a retrieval-centric agent pretrained on the target knowledge base and equipped with explicit retrieval control knobs such as constraint selection, keyword weighting, and query decomposition. SIRA follows a scalable two-stage framework that bridges retrieval and LLM inference. First, a domain-pretrained LLM produces an expected response—an expert-like sketch of what the correct answer should contain. Second, SIRA converts this expectation into a retrieval plan (keywords, constraints, and sub-queries) and executes controllable sparse retrieval that preserves fine-grained control throughout execution. This design avoids high-latency, memory-intensive vector search while tightly coupling retrieval decisions with next-token generation. Across BEIR and downstream question-answering benchmarks, SIRA consistently outperforms strong dense retrievers, SPLADE, and state-of-the-art agentic baselines, pointing to a practical path toward expert-level, controllable, and scalable retrieval-augmented agents.

1 INTRODUCTION

Information retrieval (IR) has evolved from lexical matching, exemplified by BM25 Robertson et al. (2009), to neural retrieval dominated by dense embeddings Karpukhin et al. (2020). In practice, embedding-based retrievers can work exceptionally well when trained with abundant in-domain supervision, including large-scale relevance labels and/or interaction logs that enable strong calibration 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).

However, 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. Empirically, a large-scale browsing analysis by Pew Research Center finds that when Google presents an AI-generated summary, users click standard result links substantially less often Pew Research Center (2025). This makes it increasingly difficult to *rely* on clicks as dense supervision at scale, precisely as query behavior is shifting.

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, a hallmark of conversational search. Pure similarity search is an awkward fit for this regime: dense retrieval typically exposes only a black-box nearest-neighbor operator and provides weak handles for enforcing structure (e.g., must-include/must-not-include, attribute constraints, or explicit decomposi-

tion). Neural sparse methods (e.g., SPLADE) partially restore lexical controllability while preserving learning-based ranking Formal et al. (2021), but they are still commonly used as fixed retrievers inside pipelines rather than as controllable components of an agent policy.

Yet classical lexical retrieval, exemplified by BM25, possesses underappreciated strengths that become decisive when paired with LLM reasoning. First, BM25 is *transparent*: its scoring function (term frequency, inverse document frequency, length normalization) is fully interpretable and directly manipulable. An agent can boost specific keywords, enforce must-include or must-not-include constraints, and decompose queries into sub-queries, all with predictable effects on retrieval outcomes. Second, BM25 naturally rewards *rare, discriminative terms*: high-IDF tokens receive disproportionate weight, so domain-specific jargon, acronyms, and technical phrases that would be diluted in a dense embedding become powerful retrieval signals. Third, sparse retrieval is *auditable*: one can trace exactly which keywords matched and why a document was ranked highly, a property critical for trustworthy, explainable search. Finally, BM25 avoids the latency and memory costs of maintaining large dense indices at scale. The missing ingredient has been a mechanism to surface the right rare terms and constraints, a role that LLMs, with their vast parametric knowledge, are uniquely positioned to fill.

Importantly, recent results from Google DeepMind argue that these issues are not only about missing supervision: single-vector embedding retrieval has *fundamental representational limits*. Specifically, for a fixed embedding dimension, there exist relevance structures (equivalently, sets of top-k retrieval constraints) that cannot be represented by any embedding-based nearest-neighbor scheme; the required dimension is lower-bounded by properties such as the sign-rank of the query–document relevance matrix Weller et al. (2025). As conversational interfaces induce richer compositional queries, the space of distinct constraint-satisfying outcomes can grow combinatorially, stressing the single-vector paradigm.

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 to external actions (including search) Yao et al. (2022), and recent reinforcement-learning approaches explicitly train LLMs to interleave reasoning with multi-turn web search Jin et al. (2025). Yet, in most agentic search systems, 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.

Another direction is *generative indexing*, where a model stores the corpus in its parameters and maps queries directly to doc identifiers Tay et al. (2022); Mehta et al. (2023). While appealing, purely parametric indices complicate updates and often move the retrieval interface away from explicit token-level control (e.g., interpretable constraint channels) that can be crucial for reliable, auditable search.

Goal. The central limitation of today's LLM-driven retrieval agents is not a lack of fluency or even reasoning. It is that retrieval remains a *black box*. These agents can iteratively rewrite queries and react to snippets, but they cannot directly control the retrieval process, enforce structure, or systematically exploit prior knowledge of what should exist in the corpus. As a result, they often behave like a *domain newcomer*: competent at exploratory follow-ups, yet inefficient and unreliable when faced with large, heterogeneous knowledge bases and compositional constraints. Our goal in *SuperIntelligent Retrieval Agent: The Next Frontier of Information Retrieval* is to move beyond this novice behavior and *empower retrieval agents with superintelligence*: the ability to anticipate, plan, and deliberately acquire evidence with fine-grained control, turning retrieval from an opaque tool into a programmable, auditable component of the agent's policy. Concretely, a superintelligent retrieval agent should (i) form an explicit, domain-informed hypothesis of the answer, (ii) compile that hypothesis into a structured retrieval program (constraints, weighted evidence, and decomposed sub-queries), and (iii) execute retrieval efficiently at scale—even when traditional supervision signals such as clickthrough are scarce or unreliable under answer-forward conversational interfaces.

1.1 Overview of SIRA

We propose the *SuperIntelligent Retrieval Agent (SIRA)*, a retrieval-centric agent that is pretrained on the target knowledge base and exposes explicit retrieval control knobs—constraint selection, key-

word weighting, and query decomposition to the agent policy. Unlike standard agentic search, which prompts a general-purpose LLM to "plan" using only broad pretraining and then calls an opaque retriever, SIRA couples *domain-specialized anticipation* with *mechanistically controlled retrieval*.

SIRA uses a scalable two-stage framework that bridges retrieval and LLM inference. First, we pretrain a domain model on question–corpus pairs such that, given a question that may be partially or fully answerable from the corpus, the model goes through a completely unsupervised *expertization process* to generate the text as an *expert-style explanatory response*. This expertization lets the model generate an *expected answer sketch*: a domain-grounded hypothesis of the entities, relations, and supporting facts that should appear in the correct response, mirroring how a human expert often begins with a strong prior about what the answer ought to look like. Second, conditioned on this sketch, SIRA generates a *retrieval program*: a structured plan specifying weighted keywords, optional exclusions, constraints, and decomposed sub-queries.

Crucially, the execution layer is a sparse retriever (BM25 Robertson et al. (2009)) whose mechanics are transparent to the agent. The policy is aware of the scoring structure (e.g., term-frequency contributions, inverse document frequency, and length normalization) and can therefore *engineer* retrieval queries with precise, interpretable control. We deliberately choose BM25 not despite its simplicity, but *because of it*: BM25 provides a controllable, auditable interface that an LLM can reliably manipulate. In contrast, embedding-based retrieval offers a far less interpretable control surface, where the effect of adding or removing a word on the final embedding is opaque, making constraint- and weight-level query engineering difficult to execute faithfully. Importantly, BM25's IDF weighting naturally amplifies the impact of rare, domain-specific terms, exactly the kind of expert cues that SIRA's anticipation phase surfaces. This synergy, LLM reasoning identifying discriminative terms and BM25 heavily rewarding them, bridges the historical gap between lexical retrieval's controllability and neural retrieval's semantic flexibility. The tight loop *anticipate* $\rightarrow$ *compile* $\rightarrow$ *execute* yields a practical, auditable alternative to vector search, avoiding its latency and memory costs while enabling tight coupling between retrieval decisions and next-token generation.

Across BEIR-style IR evaluation Thakur et al. (2021) and downstream question answering, SIRA, with only BM25 as the retrieval engine, consistently outperforms strong dense retrievers, SPLADE, and state-of-the-art agentic baselines. More broadly, these results suggest that domain-specialized LLM reasoning fundamentally changes the retrieval landscape: rather than treating lexical retrieval as inherently limited, we show that it can be *programmed* by an LLM into a high-performing retrieval engine. SIRA demonstrates that the perceived weakness of BM25, its reliance on exact lexical matching, becomes a strength when an LLM supplies the right vocabulary: rare expert terms that BM25 amplifies via IDF, structured constraints that BM25 can enforce precisely, and query decompositions that BM25 executes transparently. This prompts a reassessment of the assumption that embedding-based retrieval is the best path to expert-level search, and points toward a future where LLM intelligence empowers classical retrieval rather than replacing it.

2 BACKGROUND

BM25 and sparse lexical retrieval. BM25 is a widely used lexical ranking function derived from the probabilistic relevance framework Robertson et al. (2009). Given an inverted index, BM25 scores documents by aggregating per-term contributions determined by term frequency, inverse document frequency, and length normalization. Beyond efficiency, BM25 offers a transparent interface for retrieval-time control: explicit term inclusion/exclusion, interpretable term boosting, tunable normalization parameters (e.g., k_1, b), and structured execution through fielded retrieval, filters, and query decomposition. Since TF–IDF statistics are explicit and cheaply computable, BM25 provides a mechanistic substrate that supports precise, auditable manipulation of ranking behavior.

Reasoning traces and tool-using agents for search. Chain-of-Thought prompting encourages LLMs to produce intermediate reasoning steps and has been widely used for multi-step problem solving Wei et al. (2022). ReAct operationalizes this idea by interleaving reasoning with actions (e.g., tool calls) in an iterative loop Yao et al. (2022). Reinforcement learning has recently been applied to train LLMs to decide *when* to query external search engines and *how* to craft queries, as in Search-R1 Jin et al. (2025). These lines of work emphasize decision-making over multi-turn interactions where search is one component in a broader reasoning-and-action policy.

Generative and structured indexing. A complementary direction is to change the index interface itself. Differentiable/generative indexing approaches such as the Differentiable Search Index (DSI) learn to map queries directly to document identifiers, effectively storing retrieval behavior in model parameters Tay et al. (2022). Other systems build structured document indices that support traversal-style retrieval (e.g., "vectorless" or reasoning-first pipelines) Vectify AI (2025). Collectively, these approaches broaden the design space for retrieval beyond standard embedding nearest-neighbor search.

3 SUPERINTELLIGENT RETRIEVAL AGENT (SIRA)

Why SIRA. Despite their differences, most modern LLM-driven retrieval pipelines place the agent *outside* the retriever: the agent can iterate on query text and interpret returned snippets, but retrieval itself is typically treated as a fixed operator with limited direct programmability. This separation makes it hard to enforce structural constraints, exploit domain priors about what the answer should contain, or systematically control evidence acquisition under shifting query distributions and reduced click supervision. SIRA closes this gap by moving retrieval into the agent's control loop: it uses domain-specialized anticipation to form an explicit hypothesis of the answer, compiles that hypothesis into a structured retrieval program, and executes retrieval through a transparent sparse engine whose mechanics are directly manipulable.

Why BM25 as the execution layer. A natural question is why we anchor SIRA on BM25 rather than dense or neural-sparse retrieval. The answer lies in *controllability* and *synergy with LLM reasoning*. Dense retrievers compress queries and documents into fixed-dimensional vectors, but this compression obscures the contribution of individual terms: boosting a keyword, enforcing a constraint, or decomposing a query has no direct, predictable effect on the embedding. BM25, by contrast, exposes a transparent scoring surface. Its IDF component heavily rewards rare, discriminative terms; its term-frequency component rewards repetition; and its length normalization penalizes verbose documents. An LLM that understands these mechanics can *engineer* retrieval queries, selecting high-IDF domain jargon, suppressing common terms, and structuring sub-queries, with confidence that the retriever will execute the plan faithfully. Moreover, the historical limitation of BM25 was never its scoring function but its *vocabulary*: users rarely supply the rare expert terms that BM25 is designed to amplify. SIRA removes this bottleneck by using an LLM to anticipate exactly those terms, turning BM25's apparent weakness (reliance on exact lexical match) into a decisive strength.

SIRA is designed to make retrieval *programmable* for an LLM: instead of treating the retriever as an opaque tool that can only be probed via repeated query rewrites, SIRA tightly couples LLM inference with a transparent sparse retrieval engine. The core idea is to turn domain knowledge stored in model weights into explicit retrieval actions (keywords, constraints, and weights) that can be executed and audited.

We first describe SIRA's inference-time retrieval procedure, assuming the agent already behaves as a domain expert: it can anticipate plausible answers (including rare, domain-specific cues not explicitly stated in the query), propose appropriate constraints, and deliberately engineer a sparse BM25 query. We explain how SIRA acquires this expertise via purely unsupervised pretraining style *expertization* in §3.3.

3.1 INFERENCE-TIME RETRIEVAL PROCEDURE

Given a user query q, SIRA retrieves evidence through three steps: **(i) expert anticipation**, **(ii) BM25-aware plan synthesis**, and **(iii) controllable sparse retrieval execution**.

Step 1: Expert anticipation (scratchpad). In the *Expert Anticipation* phase, the domain-specialized LLM generates an internal *expected answer* E in a scratchpad. This text is not a final response; it is an expert hypothesis of what the correct answer should contain: key entities, likely mechanisms, synonymous terminology, canonical abbreviations, and "tell-tale" phrases that are common in the target corpus. Intuitively, E captures the kind of prior knowledge that an expert uses to decide what evidence to look for before issuing a search.

Step 2: BM25-aware retrieval plan synthesis. Next, SIRA conditions on (q, E) and produces a structured retrieval program π that specifies: (i) candidate keywords (including rare terms suggested

by E), (ii) query decomposition into sub-queries when appropriate, (iii) optional exclusions and constraints (e.g., `must-not` terms, metadata/field filters), and (iv) per-term weights/boosts compatible with the underlying sparse retrieval engine. To ground this step in corpus statistics, the policy can call an *IDF tool* during planning to confirm whether proposed terms are discriminative (high-IDF) and to calibrate boosts accordingly. This allows SIRA to explicitly prefer rare, domain-revealing terms when they are available, while avoiding over-weighting common terms.

Step 3: Controllable sparse retrieval execution. Finally, SIRA executes π using a BM25 retriever Robertson et al. (2009). Because the plan is expressed in the retriever's native control surface (keywords, weights, exclusions, constraints, decomposition), retrieval behavior is transparent and auditable. The top-k set D_{topk} is then consumed by the downstream answer generator to produce the final grounded response.

3.2 Example: expert tokens beyond the user query

We illustrate how anticipation yields retrieval cues that are often missing from the original query but natural for an expert.

> **User query:** "What are the recommended treatments for *H. pylori* infection after first-line therapy fails?"
>
> **Expert anticipation E (snippet):** "Likely includes *salvage therapy*, *bismuth quadruple therapy*, *levofloxacin-based* regimens, *rifabutin*, guidance to consider *antibiotic resistance*, testing methods (urea breath test / stool antigen), and timing of *test-of-cure*."
>
> **Retrieval plan π (illustrative):** *must-include*: {`salvage`, `bismuth quadruple`, `rifabutin`}
> *boost*: `levofloxacin`, `resistance`, `test-of-cure`
> *decompose*: (A) guidelines, (B) regimens, (C) resistance/testing

In this example, terms like `rifabutin` or `test-of-cure` may not appear in the user query, yet they are strong discriminators in medical corpora. Under BM25's IDF weighting, these rare terms receive disproportionately high scores, making them far more powerful retrieval signals than common words like "treatment" or "therapy." A dense retriever would blend all query terms into a single vector, diluting the discriminative power of these rare cues. SIRA's anticipation phase surfaces exactly the vocabulary that BM25 is designed to amplify, bridging the gap between LLM reasoning and classical retrieval.

3.3 Unsupervised Expertization

SIRA acquires domain expertise through a simple, fully self-supervised *expertization* pipeline over the target corpus. The expertization consists of two phases: **(i) Scratchpad (anticipation) training** and **(ii) Plan-synthesis training**. Both phases use plain next-token prediction objectives so they can be applied uniformly across large, heterogeneous knowledge bases.

No human labels, and no LLM-labeled supervision. A key design goal is to avoid any reliance on supervised relevance labels (e.g., click logs) and also avoid LLM-driven data labeling or synthetic supervision pipelines. While model-generated supervision can be effective, it often requires additional monitoring, filtering, and repeated regeneration cycles, and it can introduce training and evaluation artifacts that are difficult to detect or control Gu et al. (2024); Shumailov et al. (2024). In contrast, SIRA's expertization uses only the target corpus and simple corpus statistics, requiring no human annotation and no LLM-generated labels.

Phase I: Scratchpad (expert anticipation) training. The goal of scratchpad training is to teach the model to expand a query-like prompt into a domain-grounded "expected answer" that contains characteristic terminology, synonyms, canonical phrases, and rare cues that may not appear in the prompt itself. We construct training examples directly from the corpus using a self-supervised paragraph-completion objective.

Given a paragraph or chunk $x = (s_1, \ldots, s_m)$, we sample a sentence s_i (or a short span within s_i) as a query-like seed q and treat the full paragraph (or a surrounding context window) as the desired completion E. We then train the model with standard next-token prediction to generate E

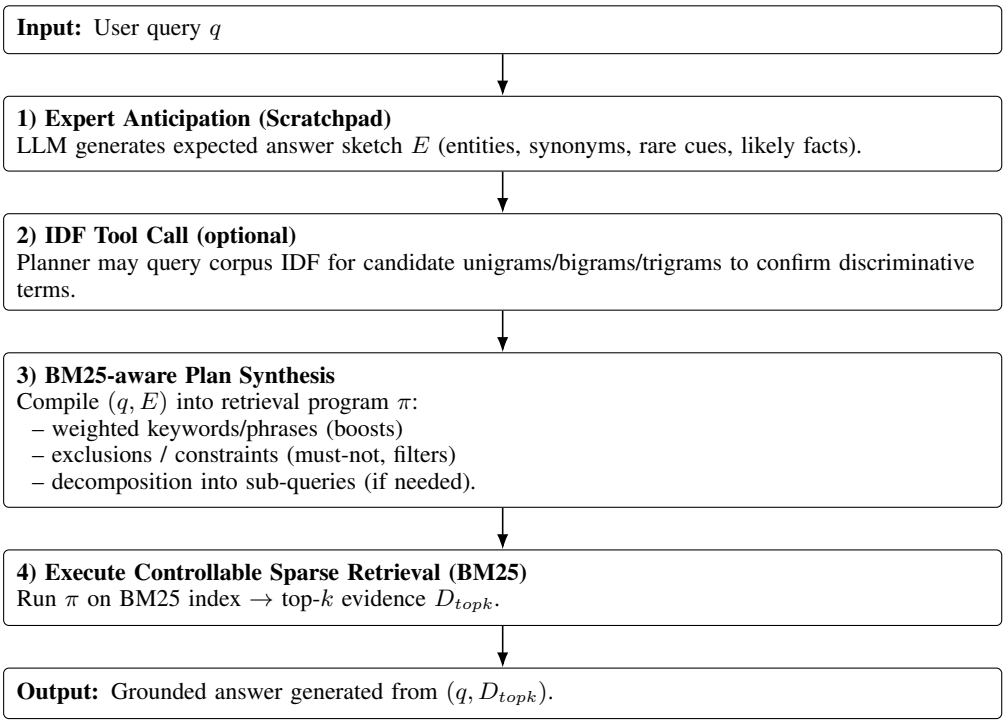

Figure 1: **SIRA inference-time retrieval.** Given a query q, SIRA first produces an expert anticipation E (scratchpad), optionally verifies discriminative terms via an IDF tool, compiles (q, E) into a structured BM25 retrieval program π (weighted keywords, constraints, and decomposition), executes controllable sparse retrieval to obtain D_{topk}, and finally generates a grounded answer using retrieved evidence.

conditioned on q:

$$\mathcal{L}_{\text{scratch}} \;=\; -\sum_{t} \log p_\theta(E_t \mid q, E_{<t})\,. \tag{1}$$

Because E is real corpus text, the model learns the domain's co-occurrence structure: which entities, phrases, and rare terms tend to appear together when a query-like cue is present. This is precisely the knowledge SIRA later uses to produce expert anticipations at inference time.

Phase II: Plan-synthesis training (BM25 anchors). The second phase trains the model to compile an anticipated answer into a sparse retrieval program that is effective under BM25-style scoring. Rather than introducing reward learning or supervised query labels, we exploit transparent corpus statistics that BM25 is known to reward. For each chunk x, we compute TF–IDF scores over unigrams, bigrams, and trigrams and select the top-K phrases (we use K=20) as anchor terms $A(x) = \{a_1, \ldots, a_K\}$. These anchors act as a lightweight, corpus-derived supervision signal for what constitutes discriminative lexical evidence.

We then train the model to generate a structured plan π, including a sequence encoding (i) weighted keywords/phrases, (ii) optional exclusions/constraints, and (iii) optional decomposition directives conditioned on the query-like seed and the scratchpad text:

$$\mathcal{L}_{\text{plan}} \;=\; -\sum_{t} \log p_\theta(\pi_t \mid q, E, \pi_{<t})\,. \tag{2}$$

At inference time, the planner may also call an IDF tool to confirm whether proposed terms are strongly discriminative and to calibrate boosts accordingly. This training directly aligns planning with BM25's mechanics: the model learns to anchor retrieval programs on rare, high-IDF cues and to express them in a form that the sparse engine can execute faithfully.

Together, scratchpad training yields domain-specialized anticipations, while plan-synthesis training teaches the model to compile those anticipations into BM25-effective keyword weights and

constraints. The key insight is that this pipeline explicitly aligns LLM generation with BM25's inductive biases: the model learns to produce exactly the rare, high-IDF terms that BM25 amplifies, and to structure retrieval programs that BM25 can execute with transparent, predictable behavior. This tight coupling, absent in systems that treat retrieval as a black-box tool, is what allows SIRA to unlock the latent power of classical lexical retrieval.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets and models. We evaluate SIRA on 11 diverse datasets spanning 6 task types to assess generalization (see Appendix A for detailed dataset statistics). Our suite includes 10 BEIR datasets Thakur et al. (2021): fact-checking (Climate-FEVER, FEVER, SciFact), citation prediction (SCIDOCS), duplicate retrieval (Quora, CQADupStack), argument retrieval (ArguAna), and QA (NQ, HotpotQA, FIQA). We also include TREC-DL 2019 to test large-scale passage retrieval (8.84M documents). For fair comparison, all agentic methods (SIRA, HyDE, CoT, Search-R1) use Qwen2.5-3B-Instruct. See Table 1 for non-agentic baseline checkpoints.

Baselines. We compare SIRA against varied retrieval paradigms: **BM25** (sparse lexical), **E5** (dense), **SPLADE** and **SPARTA** (learned sparse), **DocT5Query** (document expansion), and agentic methods including **HyDE**, **CoT**, and **Search-R1**. We also include **SIRA (w/o FT)** to quantify the gain from expertization. See Appendix B for detailed baseline descriptions.

4.2 RESULTS AND ANALYSIS

Table 1 presents the comprehensive retrieval performance across all 11 benchmarks.

Limits of lexical matching under semantic mismatch. We observe the well-known limitations of purely lexical retrieval when relevance is not expressed through surface-form overlap. BM25 is a strong lexical baseline, but it degrades on entailment- and contradiction-heavy tasks: on Climate-FEVER it achieves 0.1764 Recall@10, and on FEVER 0.6747. Here, relevant evidence often paraphrases or refutes the claim without sharing explicit n-grams, so exact matching fails.

This reflects a structural limitation rather than tuning: BM25 cannot surface evidence whose relevance is implicit rather than lexical. SIRA mitigates this while still executing over BM25 by anticipating likely answer cues (entities, mechanisms, domain terms) and injecting lexically grounded keywords into the query plan. As a result, SIRA improves Recall@10 to 0.7058 on Climate-FEVER and 0.9222 on FEVER, showing that semantic anticipation can bridge lexical mismatch without sacrificing transparent sparse retrieval.

Generalist retrievers versus domain-specialized retrieval. Dense and learned sparse retrievers such as E5 and SPLADE perform strongly as general-purpose systems, but their performance drops sharply on tasks requiring domain-specific structural knowledge. SCIDOCS illustrates this gap clearly: citation prediction depends on recognizing topical, methodological, and community-specific signals rather than semantic similarity alone. E5 and SPLADE achieve Recall@10 scores of only 0.1962 and 0.1654 respectively, whereas SIRA reaches 0.7321.

Similar trends appear in other specialized domains. On FIQA, which requires understanding financial terminology and implicit assumptions, E5 drops to 0.4697 Recall@10, while SIRA achieves 0.7935. On CQADupStack, duplicate detection depends on recognizing paraphrased intent rather than topical similarity; here SIRA achieves 0.8956 Recall@10, outperforming E5 by over 38 percentage points.

These results suggest that generalist embedding spaces struggle to adapt to task-specific relevance criteria without supervision. In contrast, SIRA internalizes domain regularities through expertization and compiles them into explicit retrieval programs, allowing it to adapt retrieval behavior across heterogeneous domains without retraining the index.

Why generic agentic reasoning fails to improve retrieval. Across multiple benchmarks, generic agentic methods such as CoT and Search-R1 fail to consistently outperform non-agentic baselines, despite producing rich intermediate reasoning traces. On FEVER, Search-R1 achieves a Recall@10

Table 1: Retrieval performance comparison on BEIR and TREC-DL 2019. All agentic methods use Qwen2.5-3B-Instruct. For non-agentic baselines, we use the official checkpoints: SPARTA (BeIR/sparta-msmarco-distilbert-base-v1), SPLADE (naver/splade-cocondenser-ensembledistil), DocT5Query (doc2query/msmarco-t5-base-v1), and E5 (intfloat/e5-base-v2). Best results in **bold**.

Method	ArguAna	Climate-FEVER	CQADupStack	FEVER	FIQA	HotpotQA	NQ	Quora	SCIDOCS	SciFact	TREC-DL 2019
Recall@10											
HyDE	0.7091	0.2598	0.3299	0.7132	0.2845	0.5051	0.4918	0.5151	0.1530	0.8344	0.1047
SPARTA	0.6181	0.1050	0.3100	0.7246	0.2450	0.5356	0.5584	0.7445	0.1303	0.7084	0.1327
DocT5Query	0.7824	0.1761	0.4339	0.6769	0.3397	0.6527	0.5068	0.8975	0.1663	0.8270	0.1456
CoT	0.7752	0.1867	0.4020	0.6765	0.3086	0.5789	0.4961	0.8704	0.1595	0.7961	0.1278
BM25	0.7738	0.1764	0.4163	0.6747	0.3198	0.6141	0.4543	0.9014	0.1636	0.8078	0.1274
Search-R1	0.7774	0.1884	0.3995	0.6587	0.3191	0.5988	0.4863	0.8609	0.1580	0.8201	0.1313
E5	0.7909	0.2899	0.5138	0.9109	0.4697	**0.7276**	0.7877	0.9428	0.1962	0.8489	0.1721
SPLADE	0.8137	0.2881	0.4924	0.8954	0.4139	0.7027	0.7381	0.9206	0.1654	0.8230	**0.1765**
SIRA (w/o FT)	0.7696	0.2265	0.3606	0.6851	0.3002	0.5689	0.4956	0.7998	0.1617	0.8512	0.1325
SIRA	**0.9403**	**0.7058**	**0.8956**	**0.9222**	**0.7935**	0.7190	**0.7935**	**0.9560**	**0.7321**	**0.9989**	0.1455
Recall@100											
HyDE	0.9531	0.4963	0.5280	0.8591	0.5408	0.6928	0.7675	0.7480	0.3640	0.9349	0.3889
SPARTA	0.8947	0.2271	0.4648	0.8488	0.4400	0.6776	0.7869	0.8954	0.3016	0.8647	0.3833
DocT5Query	0.9644	0.3739	0.5905	0.8626	0.5845	0.7879	0.7967	0.9768	0.3703	0.9177	0.4947
CoT	0.9673	0.3853	0.5742	0.8572	0.5729	0.7297	0.7861	0.9668	0.3472	0.9103	0.4620
BM25	0.9630	0.3728	0.5794	0.8609	0.5534	0.7694	0.7465	0.9768	0.3630	0.9127	0.4377
Search-R1	0.9644	0.3933	0.5658	0.8497	0.5634	0.7486	0.7752	0.9627	0.3447	0.9093	0.4565
E5	0.9737	0.5230	0.7007	0.9613	0.7204	**0.8490**	**0.9483**	**0.9941**	0.4184	0.9467	0.5385
SPLADE	0.9829	0.5209	0.6939	0.9542	0.6316	0.8176	0.9295	0.9865	0.3734	0.9320	**0.5521**
SIRA (w/o FT)	0.9659	0.4615	0.5600	0.8705	0.5415	0.7422	0.7827	0.9389	0.3748	0.9343	0.4676
SIRA	**0.9900**	**0.8685**	**0.9287**	**0.9698**	**0.8803**	0.8412	0.9041	0.9901	**0.9065**	**1.0000**	0.4382
NDCG@10											
HyDE	0.4366	0.2004	0.2463	0.5507	0.2223	0.4451	0.3315	0.3924	0.1402	0.6565	0.4949
SPARTA	0.3890	0.0852	0.2497	0.6101	0.1925	0.5132	0.3983	0.6294	0.1272	0.5894	0.6189
DocT5Query	0.4946	0.1381	0.3667	0.5122	0.2731	0.6299	0.3302	0.7960	0.1589	0.6920	0.5533
CoT	0.4951	0.1471	0.3354	0.4932	0.2486	0.5595	0.3168	0.7608	0.1518	0.6647	0.4935
BM25	0.4874	0.1372	0.3481	0.5036	0.2532	0.5851	0.2916	0.8055	0.1565	0.6791	0.4722
Search-R1	0.4911	0.1487	0.3282	0.4824	0.2530	0.5713	0.3110	0.7529	0.1511	0.6987	0.5142
E5	0.5323	0.2397	0.4196	0.8096	0.3932	**0.6905**	0.5835	0.8648	0.1855	0.7156	0.7119
SPLADE	0.5253	0.2293	0.4083	0.7933	0.3478	0.6869	0.5369	0.8344	0.1586	0.7025	**0.7352**
SIRA (w/o FT)	0.4848	0.1755	0.2786	0.5127	0.2247	0.5156	0.3249	0.6762	0.1517	0.6871	0.5649
SIRA	**0.8114**	**0.6915**	**0.8972**	**0.8444**	**0.8056**	0.6870	**0.6843**	**0.9145**	**0.7642**	**0.9982**	0.5175
NDCG@100											
HyDE	0.4904	0.2685	0.2919	0.5837	0.2889	0.4928	0.3925	0.4465	0.2117	0.6791	0.4608
SPARTA	0.4492	0.1178	0.2846	0.6372	0.2440	0.5492	0.4490	0.6656	0.1847	0.6230	0.5152
DocT5Query	0.5364	0.1929	0.4034	0.5534	0.3364	0.6645	0.3952	0.8171	0.2276	0.7133	0.5522
CoT	0.5393	0.2024	0.3750	0.5332	0.3178	0.5979	0.3810	0.7858	0.2150	0.6904	0.5020
BM25	0.5305	0.1915	0.3857	0.5447	0.3148	0.6246	0.3562	0.8257	0.2235	0.7036	0.4777
Search-R1	0.5342	0.2057	0.3665	0.5248	0.3177	0.6096	0.3748	0.7793	0.2142	0.7188	0.5060
E5	0.5734	0.3067	0.4638	0.8220	0.4610	**0.7217**	0.6217	0.8796	0.2614	0.7376	0.6571
SPLADE	0.5649	0.2965	0.4557	0.8069	0.4067	0.7164	0.5815	0.8524	0.2286	0.7248	**0.6723**
SIRA (w/o FT)	0.5296	0.2429	0.3242	0.5541	0.2877	0.5599	0.3880	0.7103	0.2239	0.7054	0.5371
SIRA	**0.8230**	**0.7427**	**0.9039**	**0.8559**	**0.8316**	0.7185	**0.7095**	**0.9244**	**0.8268**	**0.9986**	0.4954

of 0.6587 and CoT reaches 0.6765, both comparable to or worse than BM25's 0.6747. Similar stagnation appears on CQADupStack and SCIDOCS, where agentic methods cluster closely around lexical baselines. This behavior exposes a disconnect between reasoning and retrieval execution. While these agents generate semantically plausible reasoning chains, they lack explicit grounding in the statistics of the underlying index. As a result, they often propose terms that are meaningful in natural language but rare, absent, or non-discriminative in the corpus (i.e., low- or zero-IDF terms). Such terms contribute little to BM25 scoring and effectively waste retrieval capacity. SIRA addresses this failure mode by explicitly aligning reasoning with index mechanics. Through expertization, the agent learns not only which concepts are relevant, but how those concepts are expressed in the corpus. Retrieval planning is therefore constrained to operationally effective keywords, weights, and decompositions.

Regime dominance across heterogeneous retrieval tasks. SIRA achieves best-in-class performance on 9 out of 11 datasets, spanning fundamentally different retrieval logics. In argument retrieval (ArguAna), SIRA reaches 0.9403 Recall@10 compared to SPLADE's 0.8137, reflecting its ability to match counter-arguments rather than surface-level similarity. In entailment and fact-checking tasks such as Climate-FEVER and SciFact, SIRA achieves dramatic improvements, reaching 0.7058 and 0.9989 Recall@10 respectively. These gains are not confined to a single retrieval regime. On citation prediction (SCIDOCS), duplicate detection (CQADupStack, Quora), and standard QA datasets, SIRA consistently ranks relevant documents earlier and more reliably than both neural and agentic baselines, as reflected in its strong NDCG scores. This consistency indicates that

SIRA does not overfit to a specific task structure; instead, it adapts retrieval behavior by compiling domain expectations into task-appropriate control signals.

On high-resource, general-domain QA benchmarks where dense retrievers typically excel, SIRA remains highly competitive. On Natural Questions, SIRA achieves 0.7935 Recall@10 compared to E5's 0.7877, and on Quora it reaches 0.9560 versus 0.9428. On HotpotQA, SIRA remains within one point of the strongest baseline (0.7190 vs. 0.7276). These results demonstrate that SIRA's domain-specialized expertization does not come at the cost of generalization. Instead of overfitting to narrow retrieval patterns, SIRA preserves strong performance on standard benchmarks while substantially improving retrieval in domains that demand structured reasoning, semantic anticipation, or corpus-specific alignment.

On the importance of expertization. The ablation (SIRA vs. SIRA w/o FT) highlights that reasoning capability alone is insufficient without corpus alignment. Untrained agents act as *rookies*: they generate plausible-sounding queries that often fail to match the corpus's specific vocabulary (zero-IDF terms), effectively wasting retrieval slots on "hallucinated" keywords. This results in stagnation near BM25 levels (e.g., 0.1617 on SCIDOCS). Expertization converts the agent into a *domain veteran* that internalizes the index's term distribution. It learns not just *what* to ask, but *how* to ask it using valid, high-IDF discriminators. This alignment drives the massive gains, jumping from 0.2265 to 0.7058 on Climate-FEVER.

Analysis of high-density retrieval. TREC-DL 2019 presents a distinct challenge with over 200 relevant documents per query. SIRA achieves competitive but not dominant performance on this benchmark (0.1455 Recall@10 versus SPLADE's 0.1765). This reflects a deliberate design choice rather than a fundamental limitation. Standard retrievers optimize for exhaustive coverage of all topically related passages. SIRA instead prioritizes evidence sufficiency for reasoning. The agent identifies specific, high-utility documents that support coherent answer construction. This selective strategy ensures that retrieved passages align logically with the anticipated solution rather than merely matching broad topical relevance. The approach trades exhaustive enumeration for targeted, actionable evidence acquisition.

5 Conclusion

We introduced SIRA, a retrieval-centric agent that transforms retrieval from an opaque tool call into a programmable, controllable component of an LLM policy. SIRA acquires domain alignment through expertization, a process that produces corpus-grounded answer sketches and compiles them into structured retrieval programs with explicit control over term weights, constraints, and decomposed sub-queries. Across 11 diverse retrieval benchmarks, SIRA achieves the best performance on 9 of 11 datasets, with the largest gains in regimes that demand semantic anticipation and structured reasoning. Ablation experiments confirm that expertization, rather than base LLM prompting alone, drives these improvements: the expertized agent outperforms its non-expertized counterpart by +0.37 NDCG@10 on average across all datasets. These results demonstrate that strong retrieval can be achieved by compiling reasoning into index-executable lexical control, enabling sparse retrieval to rival or exceed dense and learned sparse methods while remaining interpretable and auditable.

Ethics Statement

This paper presents work whose goal is to advance the field of machine learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here. However, we note several aspects worth consideration. First, SIRA's interpretable retrieval programs enhance transparency and auditability compared to black-box neural retrieval systems. This may benefit applications where explainability is critical. Second, SIRA's computational efficiency relative to dense retrieval methods may reduce the environmental impact of large-scale information retrieval systems. We encourage future work to explore these implications further as retrieval-centric agents become more prevalent.

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.

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 Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1762–1777, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.99. URL https://aclanthology.org/2023.acl-long.99/.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *The Innovation*, 2024.

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.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 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.

Sanket Vaibhav Mehta, Jai Gupta, Yi Tay, Mostafa Dehghani, Vinh Q Tran, Jinfeng Rao, Marc Najork, Emma Strubell, and Donald Metzler. Dsi++: Updating transformer memory with new documents. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 8198–8213, 2023.

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.

Pew Research Center. 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, Hugo Zaragoza, et al. The probabilistic relevance framework: BM25 and beyond. *Foundations and trends® in information retrieval*, 3(4):333–389, 2009.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759, 2024.

Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems*, 35:21831–21843, 2022.

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.

Vectify AI. Pageindex: Document index for vectorless, reasoning-based rag, 2025. URL `https://github.com/VectifyAI/PageIndex`. GitHub repository.

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.

Orion Weller, Michael Boratko, Iftekhar Naim, and Jinhyuk Lee. On the theoretical limitations of embedding-based retrieval. *arXiv preprint arXiv:2508.21038*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 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.

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 Dataset Details

In this section we summarize the datasets used in our evaluation.

Table 2: Overview of the 11 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 8.8M 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	1.21	Retrieval for real-world Google search questions.
HotpotQA	Question Answering	7,405	5.23M	2.00	Multi-hop reasoning over Wikipedia paragraphs.
FIQA	Question Answering	648	57K	2.63	Financial QA over StackExchange data.
ArguAna	Argument Retrieval	1,406	8.67K	1.00	Matching counter-arguments for debate topics.
CQADupStack	Duplicate Question	1,570	457K	2.39	Duplicate detection across StackExchange forums.
Quora	Duplicate Question	10,000	523K	1.56	Duplicate detection for Quora questions.
SCIDOCS	Citation Prediction	1,000	25K	29.92	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	3.04	Verifying climate change claims.
SciFact	Fact-Checking	300	5K	1.13	Verifying scientific claims against abstracts.
TREC-DL 2019	Passage Retrieval	43	8.84M	215.34	Large-scale web passage retrieval.

B Baseline Descriptions

We compare SIRA against varied retrieval paradigms spanning lexical, dense, learned sparse, and agentic approaches:

Lexical Retrieval. BM25 Robertson et al. (2009) serves as the standard baseline for sparse lexical retrieval. It scores documents based on term frequency, inverse document frequency, and length normalization, providing a transparent and interpretable ranking function.

Dense Retrieval. E5 Wang et al. (2022) represents state-of-the-art dense retrieval. It encodes queries and documents into fixed-dimensional embeddings and retrieves via approximate nearest-neighbor search in the embedding space.

Learned Sparse Methods. SPLADE Formal et al. (2021) and **SPARTA** Zhao et al. (2021) perform learned term expansion and reweighting to enhance sparse retrieval. These methods use neural networks to predict importance weights for vocabulary terms, enabling them to go beyond exact lexical matching while maintaining the efficiency of sparse retrieval.

Document Expansion. DocT5Query Nogueira et al. (2019) explicitly expands documents with predicted queries. It uses a T5 model to generate likely queries for each document during indexing, enriching the document representation with additional lexical signals.

Agentic Methods. We compare against agentic methods that instruct an LLM to perform query enrichment:

- **HyDE** Gao et al. (2023) generates hypothetical documents that would answer the query, then uses these synthetic documents as retrieval queries.
- **CoT** Wei et al. (2022) produces reasoning chains to think about information needed for the query, using the reasoning trace to enrich the retrieval query.
- **Search-R1** Jin et al. (2025) uses reinforcement learning to refine queries in multi-turn interactions, learning to iteratively improve retrieval through feedback.

SIRA and Ablation. SIRA differs from these paradigms by offering *programmable retrieval*: instead of just expanding or enriching text, it couples domain-specialized anticipation with explicit control (keywords, weights, constraints) over the retrieval engine. We also include **SIRA (w/o FT)** to quantify the gain from expertization, showing the contribution of domain-specialized pretraining versus using a general-purpose LLM.

C Training Details

In this section, we provide a detailed illustration of the training process for SIRA.

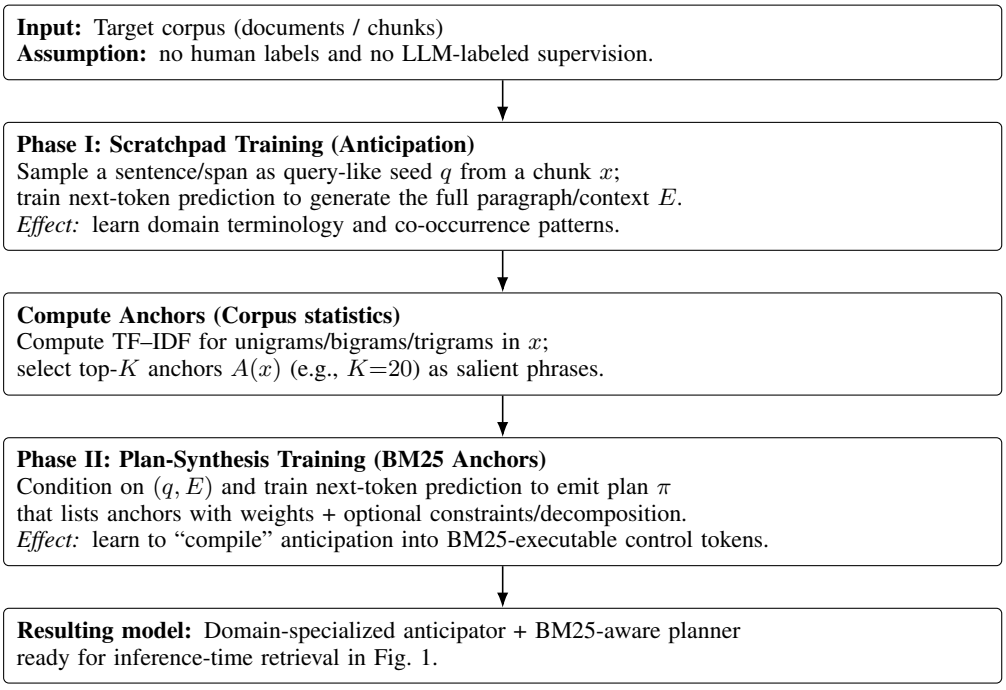

Figure 2: **Unsupervised expertization of SIRA (corpus-only).** Given only the target corpus, Phase I trains an anticipation scratchpad via self-supervised paragraph completion (seed $q \rightarrow$ context E). Phase II computes TF–IDF anchors $A(x)$ and trains the model to generate a structured BM25 retrieval program π specifying anchor weights, optional constraints, and query decomposition. Both phases use standard next-token prediction and require no human labels or LLM-generated supervision.

D Future Directions

Three promising directions emerge from this work. First, **continual test-time adaptation** could enable SIRA to refine its corpus understanding during deployment. Rather than fixing expertization after training, the agent could incrementally update its term distribution models and retrieval strategies as it encounters new queries and documents. This would support open-domain scenarios where the corpus evolves or expands over time. Second, **dynamic indexing beyond BM25** could extend programmable retrieval to richer execution backends. SIRA currently compiles retrieval plans into BM25 queries, but the same anticipation and planning framework could target agent-constructed dynamic indices that reorganize the corpus based on query patterns. Third, **multi-agent retrieval orchestration** could decompose complex information needs across specialized sub-agents. Different retrieval regimes (e.g., citation prediction, fact-checking, argument retrieval) could be handled by domain-specific expertized agents that collaborate to satisfy composite queries. SIRA represents a step toward retrieval systems with genuine expertise: agents that anticipate, plan, and deliberately acquire evidence with fine-grained control. As conversational interfaces and answer-forward search become prevalent, such capabilities will be essential for effective, interpretable, and aligned information access.