SkillQR: Agentic Skill Discovery for Reason-Intensive Query Rewriting

Anonymous ACL submission

Abstract

Dense retrievers provide grounded evidence that mitigates hallucinations in Large Language Models (LLM) generation. Real-world queries are often verbose, ambiguously framed, or require multi-step reasoning, introducing irrelevant details that weaken semantic matching for dense retrievers. Query rewriting is a widely adopted pre-retrieval technique that leverages LLMs to reformulate queries and bridge the semantic gap between such queries and the retrieval target. Existing rewriting approaches either rely on expensive proprietary models or require substantial training data, while largely applying a uniform rewriting strategy regardless of the retrieval corpus. We observe that effective query rewriting instruction is not universal: the optimal strategy depends on the retrieval target, rather than the model alone. We introduce SkillQR, a query rewriting framework that automatically discovers reasoning strategy sets from the retrieval target and dynamically routes a query to the most appropriate rewriting strategy at inference time. Without relying on training data, SkillQR achieves competitive performance compared to finetuned rewriting models on the BRIGHT and R2MED benchmarks, while also providing faster inference speed than iterative rewriting approaches.

1 Introduction

Retrieval is a core component of modern search and retrieval-augmented generation (RAG) systems (Karpukhin et al., 2020; Su et al., 2023; Wang et al., 2024; Shi et al., 2024). Retrieval quality directly determines the evidence available for downstream generation (Lewis et al., 2020; Xu et al., 2024, 2025). In real-world applications, however, user queries rarely appear in a retrieval-friendly form. Instead, they are often indirect, ambiguously framed, and expressed through surface context rather than the underlying information need (Asai et al., 2023). Such queries may contain irrelevant details or omit critical structure, which hinders effective retrieval. This challenge is especially pronounced in *reason-intensive retrieval*, where successful retrieval requires inferring the latent reasoning that connects the query to relevant evidence (Chen et al., 2025; Su et al., 2025; Shao et al., 2025). As a result, even strong dense retrievers can fail when the surface form of the query does not align with the retrieval target.

Query rewriting (Wang et al., 2023; Yao et al., 2025; Ma et al., 2023) is a natural approach to this problem. The key idea is to transform the original query into a form that better reflects the information needed for retrieval, such as implicit concepts, intermediate reasoning steps, or structural constraints. Recent work shows that such reformulation can substantially improve retrieval for complex and reasoning-intensive queries, making rewriting a practical interface between users and retrievers (Qin et al., 2025; Zhong et al., 2025; Lai et al., 2025). Despite these advantages, existing query rewriting systems remain limited in both *efficiency* and *adaptability*. Many approaches rely on expensive proprietary LLMs or on reinforcement learning (RL)-based training with substantial curated data (Chen et al., 2025; Qin et al., 2025). Others depend on iterative retrieval-and-rewrite procedures or repeated corpus interaction, which further increases inference cost and latency (Lei et al., 2025; Li et al., 2025a). As a result, current methods can be costly to build, difficult to transfer across corpora, and impractical to deploy in privacy-sensitive or resource-constrained settings.

Beyond these practical limitations, existing approaches largely keep the rewriting strategy fixed across retrieval settings, such as an instruction to "generate relevant knowledge for the query", with improvements coming mainly from stronger models or additional training. We argue that this is fundamentally incomplete: the optimal rewrite depends not only on the query, but also on the target

corpus. For example, the query "how to implement binary search" requires fundamentally different rewrites when targeting theoretical computer science papers versus Python Stack Overflow answers. However, this dependence on the *retrieval target* is largely overlooked in prior work. This motivates a corpus-aware view of query rewriting, in which **the rewriting strategy should adapt to the corpus rather than remain fixed**.

Together, these challenges highlight the need for a **lightweight and adaptive query rewriting framework** that improves retrieval through better strategy design rather than heavier models or iterative procedures. Motivated by this perspective, we propose SKILLQR, a training-free query rewriting framework that automatically discovers corpus-grounded reasoning strategies and dynamically applies them at inference time. SKILLQR extracts a compact set of rewriting strategies from the target corpus and routes each query to the appropriate strategy, enabling rewriting that is aligned with the retrieval target. Our contributions are summarized as follows:

- We show that the rewriting strategy impacts retrieval performance, and that it should be conditioned on the retrieval target.
- We propose **SkillQR**, a query rewrite framework, providing a new perspective for reason-intensive retrieval.
- We evaluate SkillQR against existing rewriting methods on two reason-intensive retrieval benchmarks, BRIGHT and R2MED, demonstrating comparable performance compared to previous methods using fewer resources.

2 Related Work

2.1 Query Rewriting for Reason-Intensive IR

The BRIGHT benchmark (Su et al., 2025) evaluates reasoning-intensive retrieval across twelve subjects, ranging from programming to natural sciences. It shows that even state-of-the-art dense retrievers remain insufficient when the reasoning gap is not addressed. R2MED (Li et al., 2025b) is a similar benchmark focusing on the medical domain content. Current research addresses reason-intensive retrieval in three ways: (1) Reasoning-aware retrievers, such as DIVER (Long et al., 2025), Reason-Embed (Chen et al., 2025), and ReasonIR (Shao et al., 2025), which utilize LLMs to generate hard negatives for contrastive learning, training retrievers to inherently bridge the reasoning gap; and (2) Reranking, where specialized ranking models refine the relevance order of retrieved outputs (Song et al., 2025b; Ji et al., 2024). (3) Our work focuses on query rewriting as a complementary approach that can be combined with advances in both dense retrieval and reranking. Representative methods include TongSearch-QR (Qin et al., 2025) and Reasoner-Rewriter (Chen et al., 2025), which use reinforcement learning or specialized fine-tuning to produce retrieval-effective rewrites.

Existing query rewriting systems suffer from several limitations. The basic approach relies on expensive proprietary models such as GPT-4o (Su et al., 2025), which hinders long-term deployment and raises privacy concerns when sensitive queries must be sent to third-party services. Training a customized query rewriter for complex query understanding is the predominant approach (Qin et al., 2025; Chen et al., 2025; Zhong et al., 2025). While more cost-effective at inference time, these approaches demand substantial training resources and carefully curated datasets, often limiting their transferability across domains. Lastly, training-free methods such as ThinkQE (Lei et al., 2025) employ pseudo-relevance feedback (PRF) by iteratively retrieving and adding top-ranked documents in LLM prompts to rewrite queries. This process increases online rewriting latency, as both the prompt length and generated output grow with each iteration, while also relying on strong reasoning LLMs, making it less practical in resource-constrained deployment settings.

2.2 Reasoning Ability of LLM Agents

LLM agents have demonstrated strong reasoning and decision-making capabilities, achieving notable performance on tasks such as mathematical problem solving and code generation (Hu et al., 2025; Zhang et al., 2024; Zhang and Xiong, 2025). Recent advances in agent-based frameworks further extend these abilities beyond pure reasoning, enabling LLM agents to interact with external tools, retrieve information from databases, and collaborate with other agents (Patil et al., 2025; Liu et al., 2026; Shi et al., 2024). Anthropic introduces agent skills (Anthropic, 2025), providing a clean and modular abstraction for agent control and context management in deployment. We adapt the skills design pattern to the problem of query rewriting.

		StackExchange							Coding		Theorem-based		
Model	**Avg**	**Bio.**	**Earth.**	**Econ.**	**Psy.**	**Rob.**	**Stack.**	**Sus.**	**Leet.**	**Pony**	**Aops**	**TheoQ.**	**TheoT.**
Original	24.4	26.2	31.4	23.5	30.1	18.1	23.9	21.0	35.0	10.4	14.9	31.6	26.7
Generic	30.4	38.4	42.4	28.8	38.9	25.8	28.5	27.4	31.7	15.3	9.7	40.2	37.5
Oracle	**33.0**	38.4	42.4	28.8	38.9	25.8	28.5	27.4	**37.2**	**30.1**	10.8	43.1	**45.0**

Table 1: NDCG@10 of different rewriting strategies on BRIGHT with ReasonIR Retriever (Shao et al., 2025). Oracle uses manually designed strategies on Coding and Theorem-based and improves retrieval on them. Generic uses a subject-agnostic rewrite strategy for all subjects. Major improvements are in Bold. All rewrites are generated with Qwen2.5-7B-Instruct.

3 SkillQR

3.1 Motivation: Rewriting Strategy

We conduct an oracle experiment on the BRIGHT benchmark (Su et al., 2025) to quantify the benefit of adopting corpus-grounded rewriting strategies. Results are shown in Table 1. Both **Generic** and **Oracle** use the same rewriting framework. However, **Generic** applies a generic strategy across all subjects, while **Oracle** employs human-curated, subject-specific strategies for Coding and Theorem-based tasks, and retains the generic strategy for StackExchange questions. We provide detailed human-designed strategies in Appendix A.3.[1]

Rewrite Instruction Template

Given a task and an input, first analyze the task and the input, then present the complete long output... (Abbreviated for presentation)

Task: *{rewrite strategy}* **Input:** *{query}*

Generic Rewrite Strategy: Generate information that is relevant and helpful to address the question in detail.

Human Strategy Example (Pony): Generate the relevant Pony syntax and language features, and provide a detailed explanation of their usage.

Table 1 shows that human-curated strategies (oracle) improve average NDCG@10 from 30.4 to 33.0 over the generic baseline, a gain of +2.6 points achieved purely through strategy adaptation with no change to the underlying model. Notably, subject Leet, where generic rewriting previously hurt performance, shows a great performance recovery with the oracle instruction. On average, the oracle demonstrates that retrieval benefits from the rewriting strategy. However, manually crafting subject-specific strategies requires a holistic understanding of the retrieval corpus that is costly to obtain and does not scale. This motivates SkillQR's approach of recovering this signal automatically via LLM agent interactions.

[1]We adopt human and generic instructions from Reasoner-Rewriter (Chen et al., 2025).

3.2 Problem Formulation

Given a query q and a document collection D, let $D_q^+ \subset D$ denote the small set of relevant documents to q. We posit that a reasoning-intensive query q is connected to D_q^+ through a latent reasoning trajectory T. This trajectory captures the implicit reasoning required to bridge the gap between the query and the relevant documents (e.g., identifying core algorithms).

The retrieval system comprises a dense retriever R and an LLM-based rewriter W_{LLM}. At inference time, the rewriter produces $q^* = W_{\text{LLM}}(q)$. The retriever R retrieves with the updated q^*, based on cosine similarity, to obtain the documents $D_{q^*}^R \subset D$. The goal of W_{LLM} is to find q^* such that:

$$\mathcal{M}(D_q^+, D_{q^*}^R) > \mathcal{M}(D_q^+, D_q^R)$$

where $\mathcal{M}$ is a ranking metric such as NDCG@10.

3.3 Agentic Offline Skill Discovery

SkillQR discovers corpus-grounded rewriting strategies through an offline agentic pipeline operating over the document collection D, requiring no training data. We provide the detailed prompts of SkillQR in Appendix A.4.

Corpus Exploration. We use the document embeddings built by the dense retriever R, thereby avoiding any additional encoding cost. The document collection is partitioned into K clusters $\{C_1, \ldots, C_K\}$, where $K \approx \sqrt{|D|}$, using a clustering algorithm in the embedding space. For each cluster C_k, we select a small set of representative documents $\tilde{C}_k \subset C_k$ closest to the cluster centroid. A proposer agent P_{LLM} characterizes each cluster using two attributes: a *domain label* and a *content-type* descriptor (e.g., domain: math, content type:

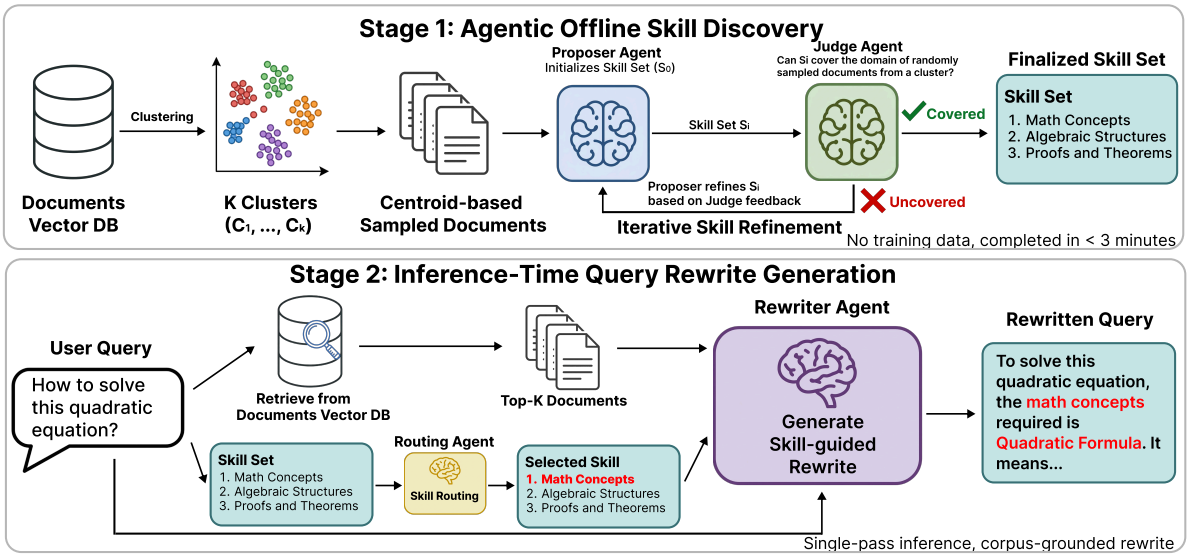

Figure 1: SkillQR has two stages: (1) Agentic offline skill discovery mines rewriting skills from the document collection; (2) the router selects a query-grounded skill for the rewriter to rewrite at inference time.

proofs). This results in a set of K cluster characterizations, denoted as $\mathcal{A} = \{(d_1, t_1), \ldots, (d_K, t_K)\}$, where each pair corresponds to the domain and content type of a cluster.

Skill Initialization. We define **skills as a set of reasoning strategies discovered from the retrieval corpus**, which are stored in the rewriter's memory and selectively applied at inference time. An LLM-based proposer P_{LLM} constructs an initial set of rewriting instructions S_0 from the cluster characterizations $\mathcal{A}$. It begins with the generic rewrite strategy in Section 3.1, denoted $\mathcal{I}$, which requires no domain expertise to craft.

Skill Initialization Prompt

Here are the groups in a collection: $\{\mathcal{A}\}$. A naive writer uses this generic skill: $\{\mathcal{I}\}$. This is too generic. Expand it to the **minimal** set of skills that covers all groups. (Abbreviated for presentation)

Iterative Skill Refinement. A judge LLM J_{LLM} validates the proposed skill set $\mathcal{S}_0$ through an iterative refinement loop. In each round i starting 0, the judge evaluates $\mathcal{S}_i$ against each cluster C_k by randomly sampling a subset of documents $\hat{C}_k \subset C_k$.

For each $\hat{C}_k$, the judge assesses whether any $s \in \mathcal{S}_i$ can guide an LLM to generate texts that are similar to documents in $\hat{C}_k$. If J_{LLM} finds such skill $s' \in \mathcal{S}_i$, then $\mathcal{S}_i$ covers $\hat{C}_k$. Otherwise, it specifies the *missing content-type* δ_k required for coverage. Formally, for each cluster C_k, the judge outputs a coverage decision:

$$J_{\text{LLM}}(\hat{C}_k, \mathcal{S}_i) \rightarrow \begin{cases} (\text{COVERED}, s'_k) & s'_k \in \mathcal{S}_i \\ (\text{UNCOVERED}, \delta_k) & \end{cases}$$

For each uncovered cluster, the judge recommends in δ_k whether it can be addressed by refining an existing skill or requires the creation of a new one. Clusters assigned to existing skills trigger targeted refinement, where the corresponding skill is broadened to cover the missing knowledge while preserving its original domain focus. Clusters deemed novel are grouped together, and the proposer generates a new skill to cover them.

Judge-Proposer Interaction Example

Judge: $\hat{C}_1$ COVERED by Skill 2 | $\hat{C}_2$ UNCOVERED δ_2: *climate dynamics* $\approx$ Skill 1 | $\hat{C}_3$ UNCOVERED δ_3: *UI/UX design* $\not\approx$ any skill

Proposer: $\hat{C}_2$ *climate dynamics* $\rightarrow$ REFINE s_1 | $\hat{C}_3$ *UX design* $\rightarrow$ NEW s_3 (Abbreviated for presentation)

Uncovered clusters are revisited in subsequent rounds and reassigned to updated skills when possible. The refinement loop terminates when all clusters are covered or a maximum number of iterations M is reached. The final skill set at termination, together with the generic instruction $\mathcal{I}$, is denoted as $\mathcal{S}_f$, which approximates the space of possible reasoning trajectories T linking queries to their retrieval targets. $\mathcal{S}_f$ serves as the final skill set for the retrieval corpus D.

3.4 Inference-Time Rewrite Generation

At inference time, rewrite agent W_{LLM} first routes the query by selecting the most appropriate skill $s^* \in \mathcal{S}_f$ for query q by evaluating the match between the query's domain and each skill in S_f. The selected skill is then used to prompt the rewriter W_{LLM} for query expansion. We provide the full routing and generation prompt in Appendix A.4.

We first perform an initial retrieval using the original query q with the dense retriever R, obtaining a set of top-ranked documents $D_R(q)$. These documents are used as pseudo-relevance feedback (PRF) for the rewriter. Conditioned on the selected skill s^* and the PRF documents $D_R(q)$, the rewriter W_{LLM} generates an expanded query guided by skill s^*. This process aims to articulate the latent reasoning trajectory T connecting the query to relevant documents. The final query is constructed as:

$$q^* = concat(q, W_{\text{LLM}}(q|D_R(q), \pi(s^*)))$$

where $\pi(s^*)$ denotes the prompt instantiated with the selected skill.

4 Experiments

4.1 Datasets and Baselines

We evaluate SkillQR on the BRIGHT (Su et al., 2025) and R2MED (Li et al., 2025b) benchmarks, reporting averaged and per-subject NDCG@10.

We compare against the following baseline: (1) ThinkQE (Lei et al., 2025) is an answer-driven query rewriting method built on QWEN-R1-Distill-14B (DeepSeek-AI, 2025), a foundation model with enhanced thinking ability. It iteratively alternates between retrieval and rewriting for three rounds. At each rewriting step, ThinkQE uses the top-5 retrieved documents as PRF to generate the answer for the query. Following the upgrade in Diver-QExpand (Long et al., 2025), we replace ThinkQE's original BM25 retriever with the dense retriever that the rewriting system supports to obtain the PRF documents. (2) TongSearch-QR (Qin et al., 2025), a Qwen2.5-7B-Instruct-based (Yang et al., 2024) query rewriter finetuned with RL on 30k curated data samples. (3) Reasoner-Rewriter (RRewriter) (Chen et al., 2025), a Qwen-based rewriter also finetuned with RL, but on 68k samples curated directly from BRIGHT documents.

To assess generalization, we evaluate all query rewriting methods on two retrievers: Diver-Retriever (Long et al., 2025) and Reason-Embed (Chen et al., 2025), both of which are leading dense retrievers on the reason-intensive retrieval tasks. We provide all model and prompt details in the Appendix A.1.

4.2 SkillQR Implementation Details

We employ QWEN-R1-Distill-14B (DeepSeek-AI, 2025) to create proposer and judge agents for all offline skill discovery processes. Skills are discovered from embeddings generated by the system's retriever (e.g., Reason-Embed or Diver-Retriever). In SkillQR, we apply MiniBatchKMeans (Pedregosa et al., 2011) to partition each subject-specific retrieval corpus into k clusters, where k is dynamically set to approximately the square root of the corpus size. We then discover skills using at most $M = 3$ refinement rounds and at most $N_S = 5$ skills. We set 3 rounds to bound offline compute cost, and cap 5 skills to ensure the skill set remains compact so routing errors reflect skill quality rather than context overload in the router. Skill discovery time is a one-time offline cost. The full BRIGHT benchmark is mined in approximately 2 hours on a single H100 GPU, with the largest subject LeetCode (400k documents) completing in 21 minutes (Appendix B). At inference time, SkillQR uses a single LLM, either Qwen2.5-7B-Instruct or QWEN-R1-Distill-14B, for both skill routing and query rewriting, requiring no additional model components beyond the chosen backbone.

4.3 BRIGHT Results

Table 2 reports SkillQR's performance on BRIGHT. At 14B-scale, SkillQR matches ThinkQE on all retrievers, reaching 34.0 and 38.5 NDCG@10 against ThinkQE's 34.1 and 39.3 on Diver and Reason-Embed, respectively. Despite comparable performance, SkillQR is approximately 40 seconds faster per query than ThinkQE, as shown in Table 3. This efficiency gain comes from SkillQR's single-pass skill-guided rewriting compared to ThinkQE's three iterative answer-generation passes.

At 7B-scale, both TongSearch and RRewriter are explicitly finetuned on training data curated with BRIGHT as the target benchmark, and RRewriter further incorporates human expert skills, introduced in Section 3.1, in its rewrite instructions. Despite this, SkillQR achieves comparable performance, reaching 38.3 on Reason-Embed against RRewriter's 38.4, while requiring less inference time than both finetuned counterparts. We attribute the inference advantage to output token length: finetuned rewriters tend to follow reasoning pat-

Method	Size	Avg	StackExchange							Coding		Theorem-based		
			Bio.	Earth.	Econ.	Psy.	Rob.	Stack.	Sus.	Leet.	Pony	Aops	TheoQ.	TheoT.
Diver-Retriever														
Original	-	31.9	48.9	50.0	32.0	41.9	30.2	28.0	33.6	13.4	16.1	8.7	37.5	42.6
ThinkQE	14B	34.1	53.1	55.7	34.9	46.3	33.3	32.9	35.9	12.5	10.5	10.6	38.9	44.6
SkillQR	14B	34.0†	54.3	54.9	34.7	45.7	33.1	32.0	36.7	10.3	11.6	9.1	40.4	44.8
TongSearch	7B	32.6	50.6	51.8	31.9	46.2	29.7	28.9	33.6	13.8	12.5	10.9	39.2	42.1
RRewriter	7B	34.5	47.5	54.8	33.7	43.7	30.0	28.0	37.2	11.3	30.2	8.9	41.1	47.8
SkillQR	7B	34.7†	51.9	52.6	32.5	47.3	33.5	31.0	37.4	11.8	22.8	9.5	39.8	46.9
Reason-Embed														
Original	-	36.2	51.8	52.5	33.5	44.8	34.3	34.6	35.7	38.2	9.3	12.8	40.8	46.3
ThinkQE	14B	39.3	61.4	55.8	33.6	49.2	32.9	40.7	40.3	34.2	16.5	15.1	42.9	48.6
SkillQR	14B	38.5†	55.9	55.0	35.1	48.5	35.5	38.1	38.0	35.9	15.6	14.6	43.2	47.2
TongSearch	7B	38.1	59.1	59.2	31.1	47.5	31.0	36.9	36.8	31.6	20.5	15.2	42.1	46.1
RRewriter	7B	38.4	55.1	59.2	32.9	44.5	32.0	37.6	35.0	31.4	23.5	16.2	45.2	48.2
SkillQR	7B	38.3†	57.7	55.7	33.0	47.9	35.3	38.5	36.5	34.8	19.2	12.8	42.2	46.0

Table 2: NDCG@10 comparison of rewriting methods on the BRIGHT. † indicates **no statistically significant difference** from the baselines of the same size (paired two-tailed t-test, $p > 0.05$).

Method	Train Data	Size	Time (s)
TongSearch	30k	7B	17.7 ± 15.2
RRewriter	68k	7B	21.1 ± 8.8
SkillQR	–	7B	13.9 ± 4.6
ThinkQE	–	14B	85.3 ± 107.1
SkillQR	–	14B	38.1 ± 22.9

Table 3: Per-query rewrite time on BRIGHT with Diver-Retriever, averaged over 240 queries (20 per subject) on a single H100 GPU.

terns observed in training data, producing redundant tokens with diminishing retrieval value when a shorter rewrite could suffice. In summary, **SkillQR's corpus-grounded skill discovery matches methods that invest heavily in domain-specific data curation and RL-based finetuning, while requiring no training data, no reward design, and lower inference cost**.

The variance of the inference time from Table 3 is also informative. We observe a large variance in ThinkQE's inference time, arising from its answer-driven strategy combined with a strong thinking model. Subjects with inherently longer answers, such as AoPS (math) and LeetCode, trigger substantially more chain-of-thought generation than subjects like psychology. In contrast, SkillQR exhibits significantly lower variance at the 14B-scale, as the corpus-grounded skill provides a focused rewriting target, bounding the generation regardless of subject complexity. We provide additional time analyses in Appendix B.

4.4 R2MED for Performance Transferability

Table 4 reports SkillQR's performance on R2MED, evaluating its transferability to the medical domain. When using the 14B thinking-enhanced model backbone, SkillQR achieves statistically comparable performance on both retrievers with less inference cost, reaching 48.9 and 49.1 NDCG@10 on Diver and Reason-Embed, respectively, against ThinkQE's 49.3 and 49.1. This confirms that SkillQR's efficiency advantage on BRIGHT transfers to the medical domain without sacrificing retrieval quality, indicating that applying corpus-grounded skills can substitute for iterative refinement even when the corpus distribution shifts substantially.

At 7B-scale, SkillQR achieves **statistically significant improvements** over TongSearch and RRewriter on two retrievers, achieving a 46.9 NDCG@10 with Diver against RRewriter's 44.9. These baselines are finetuned on synthetic data constructed with BRIGHT as their target benchmark, so the rewriters mainly learn to fill reasoning gaps present in BRIGHT's domains. When the target corpus shifts to R2MED's medical domain, those patterns no longer match the retrieval target, and the benefit of finetuning does not transfer. SkillQR avoids this failure mode by design. Rather than baking a fixed rewriting strategy into the model weights, it mines skills from whichever corpus it is deployed against, so the rewriting strategy is always aligned with the current retrieval target. **SkillQR's adaptability in the medical domain supports our central claim that rewriting**

Method	Size	Avg	Bioinfo.	Bio.	IIYi.	Medical.	MedQA.	MedXpert.	PMC-C.	PMC-T.
Diver-Retriever										
Original	-	42.7	60.3	48.9	31.3	57.6	22.5	18.5	44.0	58.5
ThinkQE	14B	49.3	69.2	54.7	33.8	61.7	36.3	23.0	51.0	64.6
SkillQR	14B	48.9†	70.1	54.1	36.0	62.7	32.9	26.5	44.9	64.3
TongSearch	7B	44.5	62.8	50.6	33.3	61.8	26.2	21.1	41.0	59.0
RRewriter	7B	44.9	64.8	47.5	31.7	62.0	28.1	23.9	40.8	60.2
SkillQR	7B	46.9*	63.5	50.3	35.5	63.4	29.5	23.0	46.1	64.0
Reason-Embed										
Original	-	42.0	62.8	51.8	22.1	61.7	25.6	21.8	38.1	51.6
ThinkQE	14B	49.1	67.7	59.8	30.9	67.8	37.7	30.4	45.6	53.2
SkillQR	14B	49.1†	70.2	56.6	34.4	65.5	34.5	31.8	46.1	55.4
TongSearch	7B	43.7	62.2	59.1	24.3	64.4	28.5	25.3	36.7	48.7
RRewriter	7B	42.3	63.5	55.1	21.7	64.0	29.2	25.7	32.7	48.7
SkillQR	7B	46.1*	68.8	58.3	26.1	64.8	30.6	27.1	37.8	55.4

Table 4: NDCG@10 comparison of rewriting methods on the R2MED benchmark. † indicates **no statistically significant difference** from ThinkQE (paired two-tailed t-test, $p > 0.05$). * indicates **statistically significant difference** from TongSearch and RRewriter (paired two-tailed t-test, $p < 0.05$).

		StackExchange							Coding		Theorem-based		
Method	Avg	Bio.	Earth.	Econ.	Psy.	Rob.	Stack.	Sus.	Leet.	Pony	Aops	TheoQ.	TheoT.
Original	31.9	48.9	50.0	32.0	41.9	30.2	28.0	33.6	13.4	16.1	8.7	37.5	42.6
Generic	32.1	48.0	51.5	30.7	43.1	28.3	28.8	36.4	12.4	18.4	9.3	37.9	40.4
Human	33.7	48.0	51.5	30.7	43.1	28.3	28.8	36.4	12.3	29.8	10.0	39.8	45.2
SkillQR	33.7	48.5	53.6	31.7	43.6	30.2	29.6	33.4	13.4	27.0	9.6	38.3	45.2

Table 5: Skill quality NDCG@10 comparison on BRIGHT with Diver-Retriever and Qwen2.5-7B-Instruct rewriter. SkillQR in this table does not inherently include PRF. SkillQR matches human expert skills on average.

strategies should adapt to the retrieval corpus rather than remain fixed, while demonstrating that such adaptation can be achieved through a lightweight, training-free framework. SkillQR also maintains consistent retrieval improvement over the original query across Diver-Retriever and Reason-Embed, suggesting that corpus-guidance is a useful signal across dense retrievers.

5 Discussions

5.1 Skill Effectiveness

Table 5 investigates the quality of automatically discovered skills. We remove the PRF documents from the rewriter's input to better isolate the effect of the skill applied. On average, SkillQR outperforms the generic rewriting strategy, yielding a 1.6 gain and matching the human-designed skills. This confirms that the discovered skills are effective and may reduce the need for manual, holistic corpus inspection. Notably, the human-designed skills default to the generic instruction for all StackExchange subjects, implicitly assuming that domain-specific strategies are unnecessary for these topics. SkillQR discovers distinct strategies across all subjects, and the resulting improvements on subsets such as Rob (+1.9) suggest that the human's assumption is incorrect. This demonstrates that SkillQR can uncover strategies overlooked by human experts. The skill effectiveness also transfers to the Reason-Embed retriever in Appendix C.3.

5.2 Skill Router Analysis

Figure 2 evaluates skill routing quality by comparing the LLM router against oracle upper and lower bounds, constructed by exhaustively applying all candidate skills per query and selecting the best and worst NDCG@10, respectively. We also include a Retriever Skill Router baseline that selects the top-1 skill via Diver-Retriever from the skill set. SkillQR's router consistently outperforms the lower bound, confirming that it makes sensible routing decisions. It matches the Retriever Skill Router on average while avoiding a separate embedding pass over the skill set. SkillQR selects the upper-bound skill for 20% of queries, exceeding the 17% random baseline over six candidates (Appendix C.2). However, a substantial gap to the upper bound re-

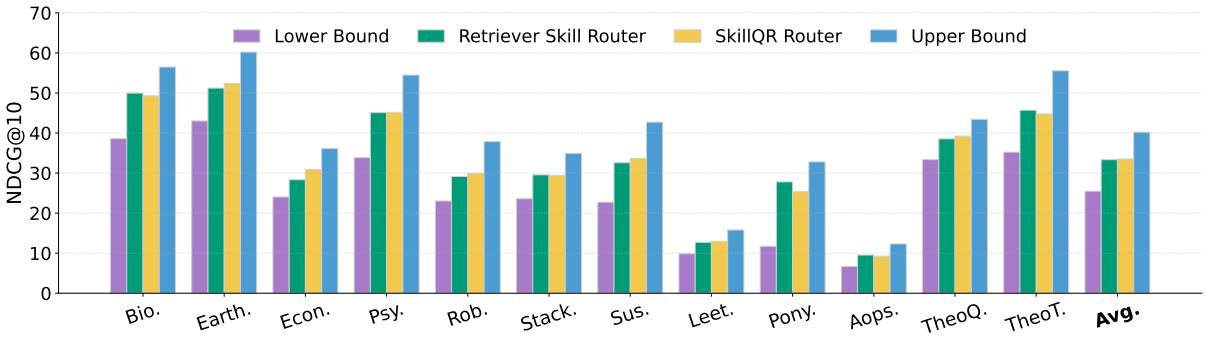

Figure 2: Analysis of SkillQR's LLM-based skill router across BRIGHT subjects.

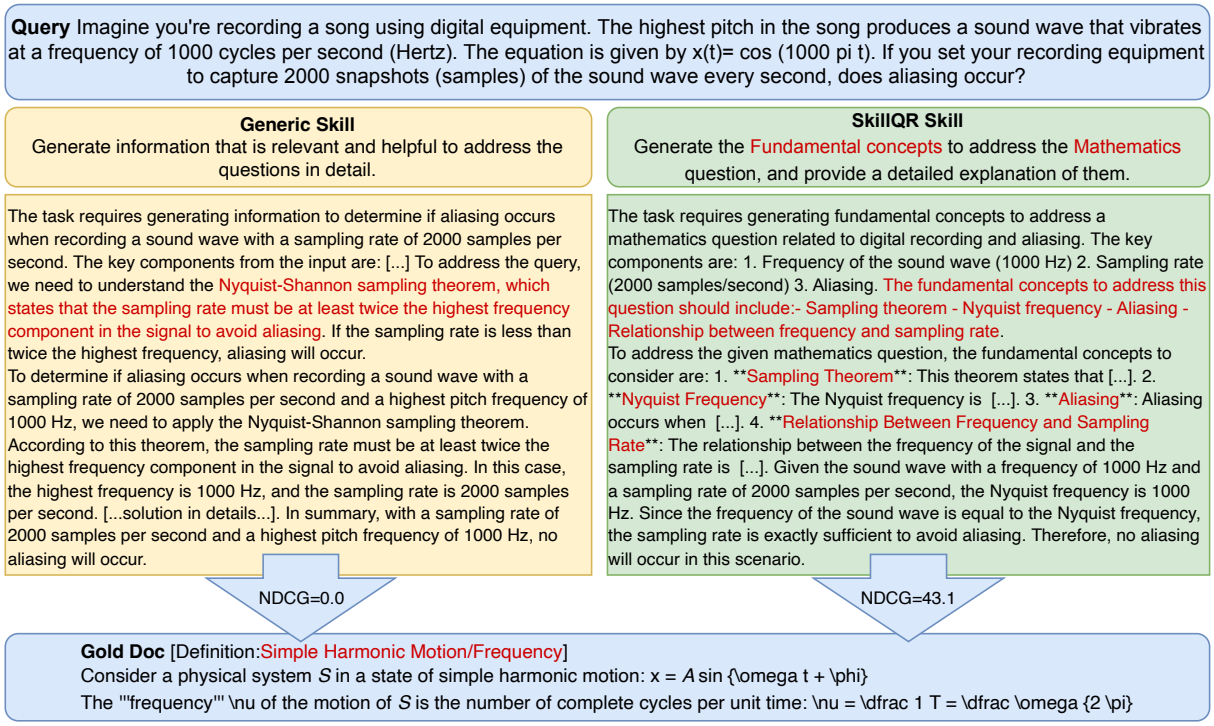

Figure 3: SkillQR directs the focus of the rewrite from problem-solving into theorem-explaining, improving the NDCG@10 score from 0.0 to 43.1. Complete rewrites are in Appendix D.

mains, suggesting that improved routing strategies could further boost performance.

5.3 Qualitative Analysis

Figure 3 presents a case where SkillQR successfully captures the relevant reasoning gap, while a generic rewrite instruction fails. The gold document is the definition of simple harmonic motion. Although the rewrite generated by the generic skill correctly identifies that answering the query requires knowledge of sampling theorems, it emphasizes solution articulation. In contrast, SkillQR understands that the corpus's concepts are most helpful to the query, and explicitly guides the model toward concept unfolding, leading the rewritten query to list relevant concepts such as the sampling theorem and Nyquist frequency. As a result, SkillQR achieves a substantially higher NDCG of 43.1.

6 Conclusion

We propose SkillQR, a training-free query rewriting framework that selects rewriting strategies from a skill set to better align queries with corpus characteristics, achieving performance comparable to strong learned baselines while maintaining a favorable efficiency–effectiveness trade-off. SkillQR highlights the retrieval corpus as an overlooked source of rewriting signal, beyond stronger models or larger training sets. We hope this corpus-conditioned perspective motivates future work on query rewriting and reason-intensive retrieval.

Limitations

While SkillQR achieves competitive performance with strong efficiency, there are limitations. SkillQR relies on LLM-based proposer, judge, and rewriter agents, whose reasoning capabilities may be affected by pretraining biases, limitations in domain knowledge, and hallucinations due to long context. Our skill router analysis also shows a gap between the LLM router and the oracle upper bound, suggesting that better query-to-skill matching could yield further gains in reason-intensive retrieval. Future versions of SkillQR may incorporate richer routing strategies and a dynamic skill refresh design for an evolving corpus. These improvements will contribute to a stronger system for reason-intensive retrieval.

References

Anthropic. 2025. Agent skills overview. Claude API-Docs.

Anthropic. 2025. Equipping agents for the real world with agent skills. Blog post.

Akari Asai, Timo Schick, Patrick Lewis, Xilun Chen, Gautier Izacard, Sebastian Riedel, Hannaneh Hajishirzi, and Wen-tau Yih. 2023. Task-aware retrieval with instructions. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3650–3675, Toronto, Canada. Association for Computational Linguistics.

Jianlyu Chen, Junwei Lan, Chaofan Li, Defu Lian, and Zheng Liu. 2025. Reasonembed: Enhanced text embeddings for reasoning-intensive document retrieval. *arXiv preprint arXiv:2510.08252*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Kaustubh D Dhole, Ramraj Chandradevan, and Eugene Agichtein. 2024. Generative query reformulation using ensemble prompting, document fusion, and relevance feedback. *arXiv preprint arXiv:2405.17658*.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. 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)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.

Yebowen Hu, Xiaoyang Wang, Wenlin Yao, Yiming Lu, Daoan Zhang, Hassan Foroosh, Dong Yu, and Fei Liu. 2025. DeFine: Decision-making with analogical reasoning over factor profiles. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 4587–4603, Vienna, Austria. Association for Computational Linguistics.

Yuelyu Ji, Zhuochun Li, Rui Meng, and Daqing He. 2024. Reasoningrank: Teaching student models to rank through reasoning-based knowledge distillation. *arXiv preprint arXiv:2410.05168*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Yilong Lai, Jialong Wu, Congzhi Zhang, Haowen Sun, and Deyu Zhou. 2025. AdaCQR: Enhancing query reformulation for conversational search via sparse and dense retrieval alignment. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 7698–7720, Abu Dhabi, UAE. Association for Computational Linguistics.

Junwei Lan, Jianlyu Chen, Zheng Liu, Chaofan Li, Siqi Bao, and Defu Lian. 2026. Retro*: Optimizing LLMs for reasoning-intensive document retrieval. In *The Fourteenth International Conference on Learning Representations*.

Yibin Lei, Tao Shen, and Andrew Yates. 2025. ThinkQE: Query expansion via an evolving thinking process. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 17772–17781, Suzhou, China. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Hang Li, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. 2025a. Llm-vprf: Large language model based vector pseudo relevance feedback. *arXiv preprint arXiv:2504.01448*.

Lei Li, Xiao Zhou, and Zheng Liu. 2025b. R2med: A benchmark for reasoning-driven medical retrieval. *arXiv preprint arXiv:2505.14558*.

Chengzhi Liu, Yuzhe YANG, Kaiwen Zhou, Zhen Zhang, Yue Fan, Yanan Xie, Peng Qi, and Xin Eric

Wang. 2026. Presenting a paper is an art: Self-improvement aesthetic agents for academic presentations. In *The Fourteenth International Conference on Learning Representations*.

Meixiu Long, Duolin Sun, Dan Yang, Junjie Wang, Yue Shen, Jian Wang, Peng Wei, Jinjie Gu, and Jiahai Wang. 2025. Diver: A multi-stage approach for reasoning-intensive information retrieval. *arXiv preprint arXiv:2508.07995*.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, and 1 others. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Xubo Qin, Jun Bai, Jiaqi Li, Zixia Jia, and Zilong Zheng. 2025. Reinforced query reasoners for reasoning-intensive retrieval tasks. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 21250–21263, Suzhou, China. Association for Computational Linguistics.

Rulin Shao, Rui Qiao, Varsha Kishore, Niklas Muennighoff, Xi Victoria Lin, Daniela Rus, Bryan Kian Hsiang Low, Sewon Min, Wen tau Yih, Pang Wei Koh, and Luke Zettlemoyer. 2025. ReasonIR: Training retrievers for reasoning tasks. In *Second Conference on Language Modeling*.

Wenqi Shi, Ran Xu, Yuchen Zhuang, Yue Yu, Jieyu Zhang, Hang Wu, Yuanda Zhu, Joyce C. Ho, Carl Yang, and May Dongmei Wang. 2024. EHRAgent: Code empowers large language models for few-shot complex tabular reasoning on electronic health records. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22315–22339, Miami, Florida, USA. Association for Computational Linguistics.

Tingyu Song, Guo Gan, Mingsheng Shang, and Yilun Zhao. 2025a. IFIR: A comprehensive benchmark for evaluating instruction-following in expert-domain information retrieval. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 10186–10204, Albuquerque, New Mexico. Association for Computational Linguistics.

Tingyu Song, Yilun Zhao, Siyue Zhang, Chen Zhao, and Arman Cohan. 2025b. LimRank: Less is more for reasoning-intensive information reranking. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20625–20639, Suzhou, China. Association for Computational Linguistics.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121, Toronto, Canada. Association for Computational Linguistics.

Hongjin Su, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han yu Wang, Liu Haisu, Quan Shi, Zachary S Siegel, Michael Tang, Ruoxi Sun, Jinsung Yoon, Sercan O Arik, Danqi Chen, and Tao Yu. 2025. BRIGHT: A realistic and challenging benchmark for reasoning-intensive retrieval. In *The Thirteenth International Conference on Learning Representations*.

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand. Association for Computational Linguistics.

Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9414–9423, Singapore. Association for Computational Linguistics.

Hui Wei, Zihao Zhang, Shenghua He, Tian Xia, Shijia Pan, and Fei Liu. 2025. PlanGenLLMs: A modern survey of LLM planning capabilities. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19497–19521, Vienna, Austria. Association for Computational Linguistics.

Ran Xu, Yuchen Zhuang, Zihan Dong, Ruiyu Wang, Yue Yu, Joyce C. Ho, Linjun Zhang, Haoyu Wang, Wenqi Shi, and Carl Yang. 2025. Acesearcher: Bootstrapping reasoning and search for llms via reinforced self-play. In *the 39th Annual Conference on Neural Information Processing Systems*.

Zhentao Xu, Mark Jerome Cruz, Matthew Guevara, Tie Wang, Manasi Deshpande, Xiaofeng Wang, and Zheng Li. 2024. Retrieval-augmented generation with knowledge graphs for customer service question answering. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 2905–2909.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 40 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Sijia Yao, Pengcheng Huang, Zhenghao Liu, Yu Gu, Yukun Yan, Shi Yu, and Ge Yu. 2025. ExpandR: Teaching dense retrievers beyond queries with LLM guidance. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 19036–19054, Suzhou, China. Association for Computational Linguistics.

Kechi Zhang, Jia Li, Ge Li, Xianjie Shi, and Zhi Jin. 2024. CodeAgent: Enhancing code generation with tool-integrated agent systems for real-world repo-level coding challenges. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13643–13658, Bangkok, Thailand. Association for Computational Linguistics.

Shaowei Zhang and Deyi Xiong. 2025. Debate4MATH: Multi-agent debate for fine-grained reasoning in math. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 16810–16824, Vienna, Austria. Association for Computational Linguistics.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Yunfei Zhong, Jun Yang, Yixing Fan, Jiafeng Guo, Lixin Su, Maarten de Rijke, Ruqing Zhang, Dawei Yin, and Xueqi Cheng. 2025. Reasoning-enhanced query understanding through decomposition and interpretation. *CoRR*, abs/2509.06544.

A Experiment Details

A.1 Models

We list the detailed model backbones for all query rewriting methods used in our experiments in Table 6. For all 7B-scale methods, we used a generation temperature of 0. For 14B-scale methods, following prior work, we used a generation temperature of 0.7 and sampled two responses per generation due to their stronger thinking ability.

A.2 Baseline Query Rewrite Prompts

We discussed various query rewriting methods in this work. Here, we list the detailed prompt instructions used in ThinkQE, TongSearch, and Reasoner-Rewriter in Figure 4.

Method	Model Backbone	Source
ThinkQE	DeepSeek-R1-Distill-Qwen-14B (QWEN-R1-Distill-14B)	(Lei et al., 2025)
TongSearch	TongSearch-QR-7B	(Qin et al., 2025)
Reasoner-Rewriter	Reasoner-Rewriter-Qwen2.5-7B	(Chen et al., 2025)
SkillQR-7B	Qwen2.5-7B-Instruct	(Yang et al., 2024)
SkillQR-14B	DeepSeek-R1-Distill-Qwen-14B	(DeepSeek-AI, 2025)
Diver-Retriever	DIVER-Retriever-4B-1020	(Long et al., 2025)
Reason-Embed	reason-embed-qwen3-4b-0928	(Chen et al., 2025)
ReasonIR	ReasonIR-8B	(Shao et al., 2025)

Table 6: Methods, backbone models, and implementation sources.

A.3 Human Skills for BRIGHT and R2MED

We present the human skills for all subjects in Figure 5. These skills are used by Oracle in Table 1, RRewriter in Tables 2 and 4, and Human Skills in Table 5.

A.4 SkillQR Prompts

Figure 6 presents the proposer agent's full prompt for skill mining, while Figure 7 shows the judge agent's prompts for skill set evaluation. Figure 9 presents the proposer's prompt for refining skill sets. The rewriter agent's full routing and generation prompt is provided in Figure 8.

B Time Analysis

Table 7 shows the offline skill mining time for each subject and its corresponding corpus sizes in BRIGHT. We can see most subjects complete skill mining within 10 mins. We also provide a subject breakdown of inference time for all methods in Table 8.

C Additional Design Choice Discussion

C.1 Clustering Algorithm

SkillQR adopts MiniBatchKMeans for its speed and stability on large corpora. We initially tested Birch clustering, but found it produced a sin-

TongSearch-QR

Instructions:
1. Identify the essential problem.
2. Think step by step to reason and describe what information could be relevant and helpful to address the questions in detail.
3. Draft an answer with as many thoughts as you have.
Query: {original query}

Reasoner-Rewriter (RRewriter)

Given a task and an input, first analyze the task and the input within the <think> and </think> tags. In your analysis:
- Break down the requirements of the task
- Identify key components from the input
- Think step by step to reason about what should be included in the output
Then, within the <response> and </response> tags, present the complete long output
Task
{subject-specific task prompt}
Input
{original query}

ThinkQE

Round 1 Instruction

Given a query and the provided passages (most of which may be incorrect or irrelevant), identify helpful information from the passages and use it to write a correct answering passage. Use your own knowledge, not just the example passages!
Query: {query}
Possible helpful passages: {top_passages_str}

Round 2 Instruction

Given a query, the provided passages (most of which may be incorrect or irrelevant), and the previous round's answer, identify helpful information from the passages and refine the prior answer.
Ensure the output directly addresses the original query. Use your own knowledge, not just the example passages!
Query: {query}
Possible helpful passages: {retrieved passage}
Prior generated answer/revised query: {round 1 generation}

Figure 4: Query Rewrite Instructions Used by Existing Query Rewriting Methods.

Subject	Size	Time (min)
Bio.	57,359	7.2
Earth.	121,249	9.5
Econ.	50,220	7.3
Psy.	52,835	7.4
Rob.	61,961	6.6
Stack.	107,081	12.7
Sus.	60,792	8.3
Leet.	413,932	21.0
Pony	7,894	4.4
Aops	188,002	16.2
TheoQ.	188,002	16.5
TheoT.	23,839	8.4

Table 7: Offline skill mining time on BRIGHT subjects using Diver-Retriever embeddings.

gle dominant cluster alongside many small fragments on retriever embeddings, leaving the proposer agent without a useful corpus overview. Re-clustering the dominant cluster with MiniBatchKMeans was required as a follow-up step after Birch, so we adopt MiniBatchKMeans directly.

C.2 Router Analysis

Following Section 5.2, Table 9 reports the router's skill selection behavior. On average, 20% of queries are routed to the oracle-best skill, exceeding the 17% random-chance baseline over six candidate skills.

C.3 Skill Performance

Table 10 shows that SkillQR's discovered skills also work with Reason-Embed, matching the skill quality trend in Section 5.1. We also evaluate an alternative skill generation strategy, SkillQR-Syn, which skips the proposer-judge aggregation. SkillQR-Syn mines skills from each cluster independently to form a skill pool. At inference time, the retriever selects the top-5 skills most semantically similar to the query, and the rewriter synthesizes a query-specific skill from these candidates. Table 10 shows that SkillQR-Syn underperforms SkillQR, suggesting that online synthesis from local skill fragments lacks the holistic corpus view

Human Defined Subject Skills (Reasoner-Rewriter)

StackExchange Subjects
Generate information that is relevant and helpful to address the question in detail.

LeetCode
Identify the required data structures and algorithms (e.g., DFS, DP, sorting, traversal), and provide a detailed explanation of how they are used to solve the problem.

Pony
Generate the relevant Pony syntax and language features (including core components), and provide a detailed explanation of their usage.

AoPS
Identify the mathematical functions and problem-solving techniques involved, and provide a detailed explanation of each.

TheoremQA_theorems
Generate the key theorem required to solve the problem, and provide a detailed explanation of the theorem.

TheoremQA_questions
Identify the mathematical concepts, formulas, and theorems required, and provide detailed explanations for each.

All R2MED Subjects
Generate information that is relevant and helpful to address the question in detail.

Figure 5: Human-Defined Rewrite Strategies for Subjects in BRIGHT and R2MED.

		StackExchange							Coding		Theorem-based		
Method	**Avg**	**Bio.**	**Earth.**	**Econ.**	**Psy.**	**Rob.**	**Stack.**	**Sus.**	**Leet.**	**Pony**	**Aops**	**TheoQ.**	**TheoT.**
ThinkQE-14B	85.3	33.7	34.3	43.6	31.1	50.6	65.9	40.6	234.3	109.8	234.5	61.3	83.4
SkillQR-14B	38.1	20.5	24.2	26.1	23.8	26.1	35.4	23.9	69.9	51.7	70.8	44.8	40.0
TongSearch-7B	17.7	18.5	14.2	17.5	22.5	22.9	13.3	14.4	24.4	25.2	156.0	9.5	14.6
RRewriter-7B	21.1	14.7	15.9	20.1	16.2	21.3	27.0	18.4	19.2	15.8	34.4	24.7	24.9
SkillQR-7B	13.9	12.2	14.2	13.2	12.6	14.3	15.5	14.0	11.9	14.3	15.8	14.5	14.1

Table 8: Per-query inference time (seconds) on BRIGHT across methods. Averages are computed over 240 queries (20 per subject).

produced by the dual-agent refinement loop.

D Additional Qualitative Analysis

We include the complete rewrite case study in Section 5.3 in Figure 10. We can see that the generic skill focuses on generating a solution to the query at hand via listing numbers and formulas, while SkillQR realizes that the core reasoning need is concept elicitation.

E AI Use Statement

Closed-source LLMs were used only to improve the clarity and presentation of the manuscript. They did not contribute to the research design, experiments, analysis, or conclusions, for which the authors take full responsibility.

SkillQR Skill Mining Prompt

You explored a document collection.
Here are the meaningful groups you found: {cluster content characterization}
A naive writer uses this generic skill: {Generic Strategy}
This is too generic. Your job: identify the MINIMAL set of skills that covers ALL meaningful groups above. Each skill tells a writer what KIND of knowledge to generate. Think:

> "If someone asks a question in this domain, what kind of knowledge should be produced to help a dense retriever locate relevant documents?"

Rules:

- Merge groups that share the same domain into one skill.
- Keep groups separate only if they require genuinely different knowledge types.
- `domain`: 1–3 words identifying the field.
- `knowledge_type`: one short umbrella term (2–5 words) covering all cluster topics.
  - Do not list topics individually.
  - Describe the knowledge itself (avoid terms like "documentation", "guides", etc.).

Return **only** a JSON object:

Figure 6: Skill mining prompt used in SkillQR to extract a minimal set of domain-specific knowledge types from clustered documents.

SkillQR Judge Prompt

A query rewriter has access to these writing instructions: {skill set}.
Below are randomly sampled documents from one cluster in the corpus.
Cluster {cluster_id} ({domain}):
{docs text}
A skill covers a cluster ONLY if it mentions the domain-specific knowledge type needed. A skill that just says "generate information" or "generate relevant content" without naming the specific knowledge type does NOT cover anything.
Judge: Do the skills above cover the type of knowledge in these documents?
Rules:
- If a skill explicitly names the knowledge type matching these documents → covered.
- If ALL skills are too generic or too vague to distinguish this cluster from others → uncovered.
- Explain what SPECIFIC knowledge type is missing from all skills.
- Even when uncovered, identify the CLOSEST skill whose domain could be broadened to cover this cluster (or null if no skill is even remotely related).
Return ONLY a JSON object:

Figure 7: SkillQR Judge prompt to evaluate the quality of the skillsets.

		StackExchange							Coding		Theorem-based		
Routing	**Avg**	**Bio.**	**Earth.**	**Econ.**	**Psy.**	**Rob.**	**Stack.**	**Sus.**	**Leet.**	**Pony**	**AoPS**	**TheoQ.**	**TheoT.**
Generic	14.7%	2.9%	4.3%	6.8%	10.9%	21.8%	30.8%	24.1%	72.5%	0.0%	0.0%	2.6%	0.0%
Gold	19.9%	13.6%	8.6%	6.8%	14.9%	24.8%	14.5%	18.5%	65.5%	38.4%	7.2%	5.2%	21.1%

Table 9: LLM router's skill selection behavior on BRIGHT. **Generic** is the percentage of queries routed to the generic skill; **Gold** is the percentage routed to the best skill for the query. The router selects the gold skill more often than the generic skill on average.

SkillQR Rewrite Generation Prompt

Routing Prompt Given a query and a numbered list of writing strategies, choose the single most fitting strategy. Do not answer the query.
Query: {original query}
Available writing strategies: {discovered skills}
Think step by step:

- What is the genre of this query?
- What knowledge is required to solve this query?
- Which strategy description matches the genre and knowledge?
- If a domain-specific strategy fits, prefer it over a general-purpose one.

Pick a strategy for the query.
Output format:
Reasoning: [reasoning]
Select: Strategy N

Rewrite Given a task and an input, first analyze them within the `<think>` and `</think>` tags. In your analysis:

- Break down the requirements of the task;
- Identify key components from the input;
- Reason step by step about what should be included in the output.

Then, within the `<response>` and `</response>` tags, present the complete output.
Task
{selected skill}
Input
{original query}
Possible helpful passages
{top-k documents retrieved by the dense retriever using the original query}

Figure 8: Routing and rewriting prompt in SkillQR.

SkillQR Refine Prompt

The judge found these clusters are close to skill {skill id} domain but NOT covered: {uncovered clusters}. Refine the instruction to also cover these clusters.
Keep the original domain focus − just broaden the {content-type} slightly.
Answer:
1. domain: The domain of this skill (1-3 words, keep the same or broaden slightly)
2. knowledge_type: Find ONE short umbrella term that covers both the original scope AND the uncovered clusters (2-5 words, no topic lists)
Return ONLY a JSON object:

The following clusters are NOT covered by any existing skill: {uncovered content-type}.
Your job: create a NEW skill instruction that covers the uncovered clusters above. The new skill must NOT duplicate what existing skills already cover.

Figure 9: SkillQR Proposer refine prompt to update skillsets. The top section is for updating an existing skill. The bottom section is to add a new skill.

		StackExchange							Coding		Theorem-based		
Method	**Avg**	**Bio.**	**Earth.**	**Econ.**	**Psy.**	**Rob.**	**Stack.**	**Sus.**	**Leet.**	**Pony**	**Aops**	**TheoQ.**	**TheoT.**
Original	36.2	51.8	52.5	33.5	44.8	34.3	34.6	35.7	38.2	9.3	12.8	40.8	46.3
Generic	36.8	51.9	53.8	30.4	47.1	31.4	34.9	37.2	34.4	24.7	13.6	41.8	40.5
Human	38.1	51.8	54.0	30.4	47.1	31.3	34.9	37.0	33.1	31.7	14.5	43.3	48.0
SkillQR	37.5	52.2	57.2	31.4	45.1	31.8	36.1	33.7	34.0	25.8	14.0	42.2	46.8
SkillQR-Syn	36.3	52.7	51.9	29.8	43.2	31.4	36.8	30.9	34.6	24.9	13.9	41.0	44.7

Table 10: Skill quality NDCG@10 comparison on BRIGHT with Reason-Embed and Qwen2.5-7B-Instruct rewriter. SkillQR methods in this table do not inherently include PRF.

Query

Imagine you're recording a song using digital equipment. The highest pitch in the song produces a sound wave that vibrates at a frequency of 1000 cycles per second (Hertz). The equation is given by x(t)= cos (1000 pi t). If you set your recording equipment to capture 2000 snapshots (samples) of the sound wave every second, does aliasing occur?

Genetic Skill

Generate information that is relevant and helpful to address the questions in detail.

The task requires generating information to determine if aliasing occurs when recording a sound wave with a sampling rate of 2000 samples per second, given that the highest pitch in the song has a frequency of 1000 Hz. The key components from the input are: 1. The frequency of the highest pitch (1000 Hz). 2. The sampling rate (2000 samples per second). 3. The need to determine if aliasing occurs. To address the query, we need to understand the Nyquist-Shannon sampling theorem, which states that the sampling rate must be at least twice the highest frequency component in the signal to avoid aliasing. If the sampling rate is less than twice the highest frequency, aliasing will occur.
To determine if aliasing occurs when recording a sound wave with a sampling rate of 2000 samples per second and a highest pitch frequency of 1000 Hz, we need to apply the Nyquist-Shannon sampling theorem. According to this theorem, the sampling rate must be at least twice the highest frequency component in the signal to avoid aliasing. In this case, the highest frequency is 1000 Hz, and the sampling rate is 2000 samples per second. Since the sampling rate is exactly twice the highest frequency, the signal is being sampled at the minimum required rate to avoid aliasing. Therefore, no aliasing will occur in this scenario. To further understand the concept, let's consider the equation given: (x(t) = cos(1000 pi t)). This represents a cosine wave with a frequency of 1000 Hz. When sampled at 2000 samples per second, each sample will capture the value of the cosine wave at regular intervals, ensuring that the original signal is accurately represented without any distortion or aliasing. In summary, with a sampling rate of 2000 samples per second and a highest pitch frequency of 1000 Hz, no aliasing will occur.

NDCG=0.0%

SkillQR Skill

Generate the Fundamental concepts to address the Mathematics question, and provide a detailed explanation of them.

The task requires generating fundamental concepts to address a mathematics question related to digital recording and aliasing. The key components are: 1. Frequency of the sound wave (1000 Hz) 2. Sampling rate (2000 samples/second) 3. Aliasing. The fundamental concepts to address this question should include:- Sampling theorem - Nyquist frequency - Aliasing - Relationship between frequency and sampling rate.
To address the given mathematics question, the fundamental concepts to consider are: 1. **Sampling Theorem**: This theorem states that a continuous-time signal can be perfectly reconstructed from its samples if the sampling rate is at least twice the highest frequency component of the signal. This minimum sampling rate is known as the Nyquist rate. 2. **Nyquist Frequency**: The Nyquist frequency is half of the sampling rate. It is the highest frequency that can be accurately represented in the sampled signal without aliasing. 3. **Aliasing**: Aliasing occurs when a signal is undersampled, meaning the sampling rate is less than the Nyquist rate. In this case, high-frequency components of the signal are incorrectly represented as lower frequency components, leading to distortion in the reconstructed signal. 4. **Relationship Between Frequency and Sampling Rate**: The relationship between the frequency of the signal and the sampling rate is crucial in determining whether aliasing will occur. If the frequency of the signal is greater than the Nyquist frequency, aliasing will occur. Given the sound wave with a frequency of 1000 Hz and a sampling rate of 2000 samples per second, the Nyquist frequency is 1000 Hz. Since the frequency of the sound wave is equal to the Nyquist frequency, the sampling rate is exactly sufficient to avoid aliasing. Therefore, no aliasing will occur in this scenario.

NDCG=43.1%

Gold Doc

[Definition:Simple Harmonic Motion/Frequency]
Consider a physical system S in a state of simple harmonic motion: x = A sin {\omega t + \phi}
The '''frequency''' \nu of the motion of S is the number of complete cycles per unit time: \nu = \dfrac 1 T = \dfrac \omega {2 \pi}

Figure 10: SkillQR directs the focus of the rewrite from problem-solving into theorem-explaining, improving the NDCG@10 score from 0.0 to 43.1.