Structured Knowledge Representation through Contextual Pages for Retrieval-Augmented Generation

Anonymous ACL submission

Abstract

Retrieval-Augmented Generation (RAG) enhances Large Language Models (LLMs) by incorporating external knowledge. Recently, some works have incorporated iterative knowledge accumulation processes into RAG models to progressively accumulate and refine query-related knowledge, thereby constructing more comprehensive knowledge representations. However, these iterative processes often lack a coherent organizational structure, which limits the construction of more comprehensive and cohesive knowledge representations. To address this, we propose PAGER, a page-driven autonomous knowledge representation framework for RAG. PAGER first prompts an LLM to construct a structured cognitive outline for a given question, which consists of multiple slots representing a distinct knowledge aspect. Then, PAGER iteratively retrieves and refines relevant documents to populate each slot, ultimately constructing a coherent page that serves as contextual input for guiding answer generation. Experiments on multiple knowledge-intensive benchmarks and backbone models show that PAGER consistently outperforms all RAG baselines. Further analyses demonstrate that PAGER constructs higher-quality and information-dense knowledge representations, better mitigates knowledge conflicts, and enables LLMs to leverage external knowledge more effectively. All codes and datasets will be available via GitHub.

1 Introduction

Retrieval-Augmented Generation (RAG) models enhance the performance of Large Language Models (LLMs) by retrieving relevant documents and using them as context inputs (Guu et al., 2020; Lewis et al., 2020). Recent studies have begun to focus on knowledge refinement to enhance RAG models (Wu et al., 2025; Xu et al., 2023b; Zhu et al., 2025), which typically incorporate specialized modules to refine retrieved documents into

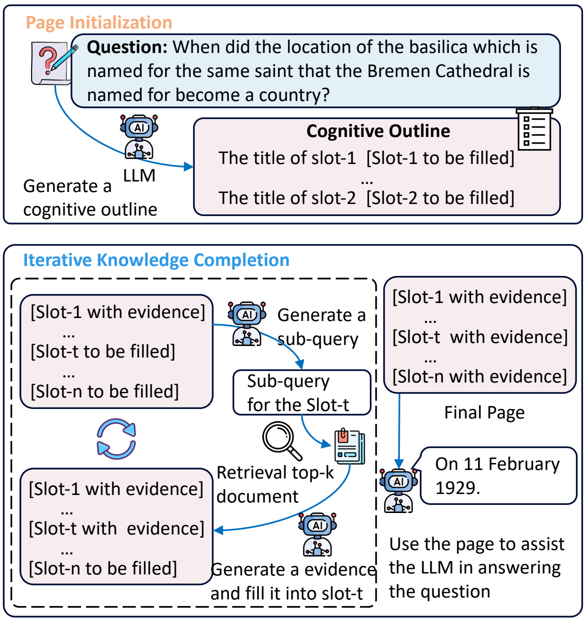


Figure 1: The Pipeline of Our PAGER Model. PAGER initializes a blank structured page with multiple slots and preliminarily fills the retrieved external knowledge into the corresponding slots, thereby constructing more comprehensive and coherent knowledge representations.

either unstructured summaries (Vig et al., 2022) or structured knowledge (Li et al., 2025b), enabling LLMs to utilize the retrieved knowledge more effectively. However, these methods primarily focus on tailoring the representation of retrieved knowledge while failing to fully leverage the structure of knowledge representations and the retriever to actively acquire and complete missing knowledge, which limits their application to complex questions (Guo et al., 2025; Jiang et al., 2025).

To achieve more effective knowledge acquisition and utilization, some studies have begun incorporating iterative knowledge accumulation processes into RAG models (Trivedi et al., 2023; Jin et al., 2025a). These methods design iterative knowledge acquisition mechanisms that leverage the strong reasoning capabilities of LLMs to progressively obtain new knowledge and refine it, thereby incorporating more information to construct more comprehensive knowledge representations (Wang et al., 2025a; Li et al., 2025a). However, these methods lack modeling of coherent knowledge structures, making it difficult for models to construct comprehensive and consistent knowledge representations to guide accurate answers. To overcome this challenge, models need to leverage their intrinsic knowledge structures to actively acquire relevant documents and construct knowledge representations aligning better with cognitive frameworks (Bruner, 1966).

In this paper, we introduce PAGER, a page-driven autonomous knowledge representation framework which constructs contextual pages for RAG modeling, allowing LLMs to leverage their reasoning and planning capability to organize and exploit information more effectively. Specifically, as shown in Figure 1, PAGER first prompts the LLM to draw on its parametric knowledge to construct a structured cognitive outline for the target question. This outline consists of multiple slots, each representing a distinct aspect of the potentially relevant knowledge needed to answer the question. Then PAGER employs an iterative knowledge completion mechanism to iteratively retrieve supporting documents for each slot, refine them into concise knowledge evidence, and fill the corresponding slot in the page. This iterative process continues until all slots are filled with the corresponding knowledge evidence. Finally, PAGER uses this structured page as contextual knowledge to guide the LLM to answer the given question.

Experimental results demonstrate that PAGER consistently outperforms all baseline methods on knowledge-intensive tasks of varying scenarios and different backbone models, demonstrating its effectiveness. Further analysis shows that the cognitive architecture designed by PAGER can more effectively guide LLMs to acquire and organize knowledge, yielding coherent and comprehensive knowledge representations that support question answering. Furthermore, compared to other knowledge representations, the structured knowledge pages exhibit richer information content and superior quality, demonstrating the effectiveness of the knowledge representations constructed by PAGER. Meanwhile, by organizing external knowledge into structured pages, PAGER effectively mitigates knowledge conflicts within LLMs and enables them to utilize the knowledge more effectively.

2 Related Work

Large language models (LLMs) (Yang et al., 2025; Dubey et al., 2024) have demonstrated strong capabilities across a wide range of tasks (Trivedi et al., 2023; He et al., 2021). However, LLMs typically suffer from hallucination, which can lead to incorrect responses (Jiang et al., 2023; Xu et al., 2023a). To mitigate this issue, existing studies employ Retrieval-Augmented Generation (RAG) models, which retrieve relevant documents for a given question and incorporate them as input context, enabling LLMs to access external knowledge and generate accurate answers (Lewis et al., 2020; Guu et al., 2020). However, the conflict between retrieved knowledge and parametric memory hinders the LLM's ability to reliably identify critical facts from retrieved documents, limiting the effectiveness of RAG models (Huo et al., 2025).

To address these challenges, some studies have focused on refining knowledge to enhance LLMs' ability to capture crucial evidence from retrieved documents (Wu et al., 2025). Some studies use query-focused summarization methods (Xu et al., 2023b) to condense retrieved documents into shorter forms, thereby reducing noise and enhancing relevance. However, the generated summaries may fail to capture the relationships among retrieved documents, leading to the loss of supporting evidence that is distributed across these documents. Furthermore, several studies have begun exploring the transformation of retrieved documents into structured knowledge, such as graphs (Zhu et al., 2025) or tables (Li et al., 2025b), thereby better capturing knowledge from different documents. Nevertheless, these approaches fail to leverage structural forms to complete the knowledge representation, despite the availability of retrieval tools.

Recently, some studies have shifted toward acquiring and accumulating additional knowledge to enhance the process of refined knowledge representation (Trivedi et al., 2023). These methods typically iteratively generate sub-queries to target existing knowledge gaps and retrieve relevant documents, and then apply different knowledge refinement strategies to construct diverse refined knowledge representations (Wang et al., 2024, 2025a). For example, Wang et al. (2025a) refine knowledge into note-based representations, and Li et al. (2025a) adopt a reasoning-in-document mechanism that integrates knowledge into the reasoning trajectory. However, these methods fail to effectively model coherent knowledge structures, which are essential for constructing more comprehensive and cohesive knowledge representations (Bruner, 1966). To overcome this limitation, PAGER leverages structural pages to integrate more relevant knowledge and organize it into a more cognitively structured format.

3 Methodology

In this section, we introduce PAGER, a page-driven autonomous knowledge searching, accumulation and representation framework. We first describe the preliminaries of RAG models enhanced with structured knowledge representations (Sec. 3.1) and then describe the construction of coherent and comprehensive knowledge representations via autonomous page construction (Sec. 3.2).

3.1 Preliminaries: Enhancing RAG Models with Structured Knowledge

Recent RAG approaches incorporate structured knowledge representations to refine retrieved information (Li et al., 2025b). Different from them, PAGER aims to guide an LLM to iteratively retrieve relevant documents and complete knowledge representations by autonomously interacting with retrieval tools, thereby enabling more accurate and reliable query answering.

Knowledge Refinement with Structured Representations. Given a query q, existing methods (Li et al., 2025b) first retrieve the top-k relevant documents:

$$D = \text{Search}(q, k), \tag{1}$$

where $D = \{d_1, \ldots, d_k\}$. Subsequently, the LLM is prompted to refine the retrieved documents into a query-conditioned structured knowledge representation result O. The refined knowledge O can take various forms, such as tables, graphs, or other structured abstractions. The structured knowledge O is then provided as contextual input to the LLM for answer generation:

$$y = \mathcal{M}\big(\text{Instruct}_{\text{QA}}(O, q)\big), \tag{2}$$

where $\mathcal{M}$ indicates the LLM and $\text{Instruct}_{\text{QA}}$ denotes the instruction template that guides the LLM to produce an answer grounded in the structured knowledge. Despite their effectiveness, these approaches primarily treat structured knowledge as a post-retrieval refinement artifact. They do not fully exploit structured representations as an active medium for completing knowledge, even when external retrieval tools are available.

Knowledge Completion Mechanism. In contrast to prior methods that rely on static structures for knowledge refinement, we propose PAGER, which introduces a cognitively inspired *page* structure to facilitate knowledge completion for the questions and obtain a structured page p:

$$p = \mathcal{F}_{\text{page}}\big(q, \text{Search}(\cdot)\big), \tag{3}$$

where $\mathcal{F}_{\text{page}}$ denotes a page construction function that autonomously accumulates and organizes knowledge through iterative interactions with retrieval tools, as detailed in Sec. 3.2. The LLM then performs reasoning over the resulting structured page p to derive the final answer y:

$$y = \mathcal{M}\big(\text{Instruct}_{\text{QA}}(p, q)\big). \tag{4}$$

3.2 Knowledge Representation via Structured Page Construction

In this subsection, we provide a detailed description of the structural contextual pages for RAG models. We first introduce the page initialization mechanism in PAGER, and then describe how PAGER iteratively completes knowledge to construct a structured page.

Page Initialization. Given a query q, we initialize the page solely based on the intrinsic reasoning and planning capabilities of the LLM, without incorporating any retrieved external knowledge.

Specifically, we design an instruction template $\text{Instruct}_{\text{outline}}$ to prompt the LLM to analyze the query q and generate a coherent reasoning trace r. Conditioned on this reasoning process, the model then constructs an initial page p_0:

$$r, p_0 = \mathcal{M}\big(\text{Instruct}_{\text{outline}}(q)\big). \tag{5}$$

The construction of p_0 is guided by the continuous reasoning trace r, which captures the LLM's internal analytical process and semantic understanding of the query q. Nevertheless, such a reasoning-driven outline remains abstract and may lack concrete supporting evidence (Wang et al., 2024), motivating subsequent retrieval and knowledge accumulation to progressively fill these blank slots. Thus, the initial page p_0 can be formulated as a structured cognitive outline comprising n explicitly defined blank slots, which are reserved for subsequent knowledge completion:

$$p_0 = \text{Outline}(\texttt{[b]}_1, \ldots, \texttt{[b]}_n), \tag{6}$$

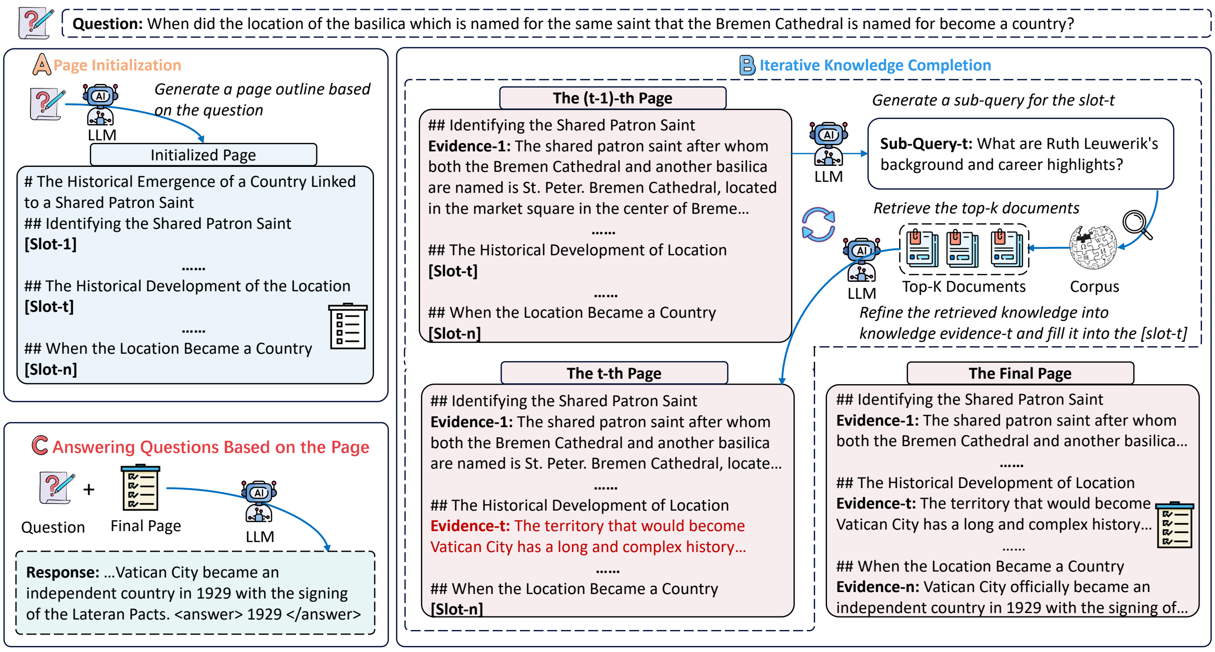


Figure 2: The Illustration of Our PAGER Model.

where $\texttt{[b]}_i$ denotes the i-th placeholder corresponding to a specific knowledge component required to answer the query. Through this process, PAGER establishes a high-level cognitive structure for the query, decomposing it into n thematically organized knowledge slots that specify the necessary background and supporting information.

Iterative Knowledge Completion. PAGER then performs an iterative knowledge refinement process to fill the missing slots $\texttt{[b]}_{1:n}$.

At each iteration step t ($1 \leq t \leq n$), the $(t-1)$-th page is represented as:

$$p_{t-1} = \text{Outline}(s_1, \ldots, s_{t-1}, \texttt{[b]}_t, \ldots, \texttt{[b]}_n). \tag{7}$$

PAGER obtains the page p_t by updating the $(t-1)$-th page p_{t-1} through filling the t-th slot $\texttt{[b]}_t$:

$$s_t \xrightarrow{\text{fill}} \texttt{[b]}_t, \tag{8}$$

where s_t is a knowledge evidence used to fill the missing slot $\texttt{[b]}_t$. Through this progressive iterative mechanism, external knowledge is gradually refined and incorporated into the knowledge pages until all n slots are filled with supporting evidence, yielding the final structured page p_n.

To gather supporting evidence s_t for the slot $\texttt{[b]}_t$, we first prompt the LLM to generate a sub-query q_t tailored to gather necessary supporting evidence to fill the t-th slot $\texttt{[b]}_t$ in p_{t-1}:

$$q_t = \mathcal{M}(\text{Instruct}_{\text{query}}(p_{t-1}, q)), \tag{9}$$

where $\text{Instruct}_{\text{query}}$ is an instruction template designed to guide the LLM to focus on the blank slot $\texttt{[b]}_t$ and produce a sub-query for retrieving relevant topical knowledge based on the current page p_{t-1} and the original query q. Next, the retriever model is employed to search k documents D_t based on the generated query q_t:

$$D_t = \text{Search}(q_t, k), \tag{10}$$

and finally, the LLM is prompted using $\text{Instruct}_{\text{fill}}$ to generate the knowledge evidence s_t based on the retrieved documents D_t:

$$s_t = \mathcal{M}(\text{Instruct}_{\text{fill}}(q, p_{t-1}, D_t, q_t)). \tag{11}$$

4 Experimental Methodology

In this section, we describe the datasets, evaluation metrics, and baselines, followed by the implementation details of our experiments. More experimental details are provided in Appendix A.2.

Dataset. Following previous work (Li et al., 2025a; Song et al., 2025), we evaluate our method on both multi-hop and single-hop QA benchmarks. Specifically, we select HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023) for multi-hop tasks, while using NQ (Kwiatkowski et al., 2019) and AmbigQA (Min et al., 2020) for single-hop tasks. For evaluation, we randomly sample 2,000 instances from the development set of each dataset, except for Bamboogle (Press et al., 2023), where we utilize the entire test set (125 instances) due to its limited size.

Evaluation Metrics. We follow existing work (Sun et al., 2025; Song et al., 2025) to utilize Cover Exact Match as the evaluation metric.

Baselines. In our experiments, we compare PAGER with multiple baseline models, including a Vanilla LLM, one-pass retrieval RAG models, iterative retrieval RAG models, and iterative knowledge representation construction RAG models.

For the Vanilla LLM, we directly feed the query to the LLM and ask it to generate the answer without any external knowledge. For one-pass retrieval RAG models, we adopt Vanilla RAG and StructRAG (Li et al., 2025b), where the former utilizes retrieved documents as contextual input to assist the LLM to answer the question, and the latter designs a router to refine documents into structured knowledge representations as input context. For iterative retrieval RAG models, we employ Iter-RetGen (Shao et al., 2023) and IRCoT (Trivedi et al., 2023) to accumulate knowledge for assisting the LLM. These methods interleave retrieval with the generation process, utilizing intermediate generated content to guide subsequent retrieval and directly leveraging the retrieved documents to facilitate the generation process. Besides, we adopt RAT (Wang et al., 2024), Search-o1 (Li et al., 2025a), and DeepNote (Wang et al., 2025a) as iterative knowledge representation construction RAG models, which not only accumulate knowledge but also refine it into distinct forms of knowledge representations. Specifically, RAT directly prompts the LLM to generate CoT-style knowledge representations and iteratively retrieves external knowledge to refine and revise each reasoning step. Search-o1 performs adaptive retrieval during reasoning and applies a Reason-in-Document mechanism to refine the retrieved documents, integrating them into the CoT-style representation. DeepNote compresses retrieved knowledge into note-based representations and iteratively acquires additional information to update these notes, ultimately constructing an optimal note to assist the LLM.

Implementation Details. In our experiments, we employ Qwen3-32B[1] (Yang et al., 2025), and Llama-3.1-70B-Instruct[2] (Dubey et al., 2024) as backbone models. We follow FlashRAG (Jin et al., 2025b) to use Wikipedia as the retrieval corpus and adopt Qwen3-Embedding-0.6B (Zhang et al., 2025) as the embedding model and employ FAISS (Johnson et al., 2019) to build indexes for retrieval. During retrieval, the top-5 ranked documents are retained for all RAG models. We further employ the vLLM inference framework (Kwon et al., 2023) to accelerate inference across all models.

[1]https://huggingface.co/Qwen/Qwen3-32B

[2]https://huggingface.co/meta-llama/Llama-3.1-70B-Instruct

5 Experimental Results

In this section, we first evaluate the performance of PAGER across different models and datasets. Subsequently, we conduct ablation studies to analyze the effectiveness of different functional components in PAGER. Then, we investigate the effectiveness of the constructed pages in knowledge representation. Finally, we explore the impact of different representations on knowledge utilization in LLMs.

5.1 Overall Performance

As shown in Table 1, we compare the overall performance of PAGER with various baseline methods across a range of knowledge-intensive tasks.

Overall, PAGER demonstrates its effectiveness by outperforming all baseline models, achieving improvements exceeding 2%. Notably, PAGER consistently shows improvements across various tasks and backbone LLMs, underscoring its robust generalization ability. When compared to StructRAG, PAGER delivers an average performance boost of over 5%, indicating that the page format serves as an effective knowledge representation mechanism, enabling autonomous interaction with external knowledge to build more comprehensive knowledge structures. Furthermore, in comparison to iterative retrieval-based methods like IRCoT and Iter-RetGen, PAGER achieves an improvement of over 9%, emphasizing its role in more effectively refining and organizing retrieved evidence. Additionally, PAGER surpasses DeepNote, which facilitates better integration of retrieved knowledge by summarizing them as notes, suggesting that the construction of structured knowledge pages allows PAGER to generate more effective knowledge representations, thus providing stronger support for LLMs in answering questions.

5.2 Ablation Study

In this subsection, we conduct ablation studies to evaluate the effectiveness of different components

Methods	HotpotQA	2WikiMQA	MuSiQue	Bamboogle	NQ	AmbigQA	Avg.
Qwen3-32B							
Vanilla LLM	28.6	31.8	8.2	44.0	36.2	32.9	30.3
Vanilla RAG	42.9	36.5	13.5	52.0	54.6	54.8	42.4
StructRAG (2025b)	42.5	31.5	12.8	44.8	55.1	54.5	40.2
IRCoT (2023)	45.1	45.5	14.7	32.0	52.2	51.8	40.2
Iter-RetGen (2023)	43.2	40.2	14.5	37.6	**56.5**	55.4	41.2
RAT (2024)	43.7	45.0	16.1	55.2	53.2	55.5	44.8
Search-o1 (2025a)	47.6	47.0	22.9	33.6	49.2	51.5	42.0
DeepNote (2025a)	48.4	47.2	17.2	39.2	55.6	55.9	43.9
PAGER	**50.6**	**57.4**	**23.0**	**62.4**	**56.5**	**56.4**	**51.1**
Llama3.1-70B-Instruct							
Vanilla LLM	37.7	41.5	14.1	57.6	50.3	48.8	41.7
Vanilla RAG	48.2	41.9	19.1	56.8	55.7	57.0	46.5
StructRAG (2025b)	49.6	44.1	19.4	57.6	**57.4**	58.2	47.7
IRCoT (2023)	35.2	37.0	6.2	11.2	49.9	50.3	31.6
Iter-RetGen (2023)	41.1	34.5	11.9	32.8	53.0	52.3	37.6
RAT (2024)	48.9	40.2	20.8	58.4	53.6	56.4	46.2
Search-o1 (2025a)	50.2	**55.4**	22.9	58.4	51.6	53.8	48.3
DeepNote (2025a)	51.7	48.9	20.8	58.4	56.6	58.4	49.1
PAGER	**52.4**	54.9	**24.3**	**62.4**	56.4	**60.0**	**51.7**

Table 1: Overall Performance of Different RAG Models. The **best** and second best results are highlighted.

Methods	HotpotQA	2WikiMQA	MuSiQue	Bamboogle	NQ	AmbigQA	Avg.
Qwen3-32B							
PAGER (Parallel Filling)	45.9	43.8	18.8	59.2	56.4	**57.0**	46.9
PAGER	**50.6**	**57.4**	**23.0**	**62.4**	**56.5**	56.4	**51.1**
w/o IterRetrieval	44.8	42.2	16.9	52.0	54.6	55.1	44.3
w/o Initilization	46.2	45.2	15.9	48.0	55.2	55.9	44.4
Llama3.1-70B-Instruct							
PAGER (Parallel Filling)	47.4	43.4	18.6	58.4	**57.2**	59.5	47.4
PAGER	**52.4**	**54.9**	**24.3**	**62.4**	56.4	**60.0**	**51.6**
w/o IterRetrieval	47.6	41.6	18.1	56.0	56.3	58.1	46.3
w/o Initilization	51.1	50.5	21.4	60.0	55.9	57.9	49.5

Table 2: Ablation Study. The **best** and second best results are highlighted.

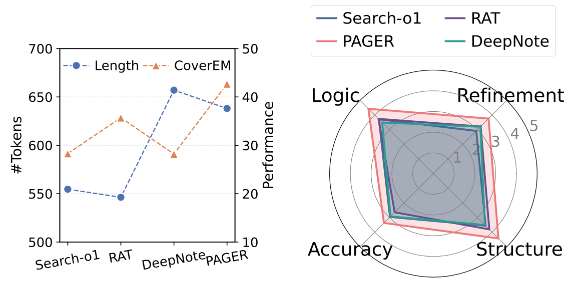
in PAGER.

In the experiments, we compare PAGER with three ablated models: PAGER (Parallel Filling) simultaneously generates sub-queries for all missing slots in the initial page, conducts parallel retrieval and then refines these retrieved documents to fill into the corresponding slots; PAGER (w/o IterRetrieval) conducts a single-pass retrieval based on the given query and filling the intial pages based on these retrieved docuemtns; PAGER (w/o Initialization) iteratively refines the retrieved documents into concise summaries, concatenating them until the aggregated summaries are sufficient to answer the question. This variant removes the page initialization stage, which typically generates the outline based on the internal planning ability of LLMs.

As shown in Table 2, PAGER consistently outperforms PAGER (Parallel Filling), demonstrating the effectiveness of the iterative knowledge completion mechanism in PAGER. Unlike parallel retrieval, PAGER generates queries conditioned on the page state at the previous step, which enables more tailored queries for retrieving necessary information and helps avoid homogeneous content to fill in different page slots. Furthermore, compared to PAGER (w/o IterRetrieval), PAGER exhibits consistent performance gains across different datasets and backbone models, indicating that iterative retrieval can incorporate more essential information to better answer the given query. Finally, compared with PAGER (w/o Initialization), PAGER achieves further improvements, highlighting the critical role of the cognitive outline generated by leveraging the reasoning capability of LLMs. This outline enables PAGER to effectively organize and complete the required knowledge within a page.

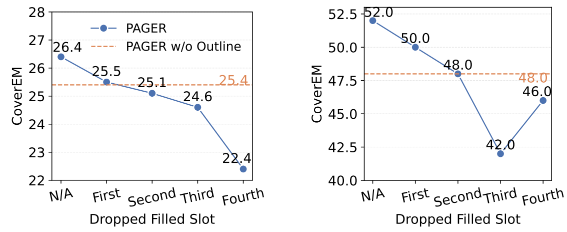
5.3 The Effectiveness of Constructed Pages in Knowledge Representation

In this section, we investigate the effectiveness of PAGER in knowledge representation by analyzing its constructed pages. Specifically, we examine both the quality of the constructed pages produced by PAGER and how these pages facilitate knowl-



(a) Representation Length. (b) GLM Score.

Figure 3: The Quality of Knowledge Representations Constructed by Different Methods.



(a) MuSiQue. (b) Bamboogle.

Figure 4: Slot Ablation Studies on the Constructed Page. "N/A" denotes the complete page with no filled slots removed. "First", "Second", "Third", and "Fourth" denote the variants in which the First, Second, Third, and Fourth filled slots are removed, respectively. PAGER (w/o Outline) denotes a variant where the cognitive outline structure is removed from the complete page and is used as a baseline for comparison.

edge utilization by LLMs. In this experiment, we adopt Qwen3-32B as the backbone model and conduct evaluations on the MuSiQue and Bamboogle.

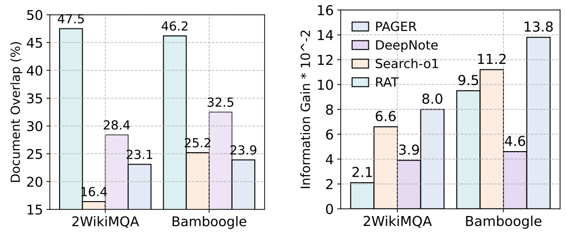
The Quality of PAGER in Representing Retrieved Knowledge. Figure 3 presents the length statistics and quality assessments of the constructed knowledge representations.

As shown in Figure 3(a), we compute the average token length of knowledge representations generated by different methods to analyze the trade-off between the length and QA performance. Compared with RAT and Search-o1, PAGER achieves superior performance with longer knowledge representations, indicating that PAGER enhances QA performance by incorporating richer information to form more comprehensive knowledge representations. Compared with DeepNote, PAGER attains significantly better performance while reducing the length of the knowledge representation. This suggests that PAGER exhibits higher knowledge density than DeepNote, enabling it to deliver more effective information within the constructed page.

We further evaluate the overall quality of the knowledge representations generated by different models. As shown in Figure 3(b), we employ a strong closed-source LLM, GLM-4.5, as the evaluator. Using the prompt templates provided in Appendix A.9, we assess the knowledge representations along four dimensions: accuracy, logicality, structure, and degree of knowledge refinement, with each dimension rated on a scale from 0 to 5. The evaluation results demonstrate that PAGER consistently outperforms other models across all dimensions, with particularly notable advantages in structure and logical consistency. These findings further validate the effectiveness of the pages constructed by PAGER, which receive higher scores from the GLM evaluator, reflecting the coherent organization and comprehensive coverage of the retained knowledge.

Effectiveness of Completed Knowledge in Different Slots of the Page. As shown in Figure 4, we further analyze the effectiveness of the knowledge filled in different slots of the page. We collect pages containing four slots as seed pages (As shown in Appendix A.7, the pages constructed by the Qwen3-32B predominantly feature 4 slots). Then, we remove the first, second, third, and fourth filled slots to construct incomplete pages, respectively. Finally, we feed these four distinct incomplete pages into the model to evaluate the performance.

Overall, compared with the completed page PAGER (N/A), removing any single filled slots from the page leads to performance degradation, indicating that all filled knowledge is necessary to support LLMs in answering the query. As the removed filled slot shifts from the first slot to the fourth slot, the model performance degrades more substantially, suggesting that knowledge filled in later slots is more critical for question answering. One possible reason is that our iterative completion method tends to acquire increasingly necessary knowledge in later slots, which aligns with the reasoning process of LLMs, where later reasoning steps often involve deeper inference to answer the query. Furthermore, compared with PAGER (N/A), PAGER (w/o Outline) also exhibits reduced performance when the removed knowledge is from slots later than the second one. This observation further indicates that the cognitive structure plays a critical role in guiding the knowledge construction process, thereby verifying the effectiveness of our



(a) Document Overlap.

(b) Information Gain.

Figure 5: Effects of Different Knowledge Representations on Retrieval and Generation Modules of RAG.

Methods	2WikiMQA	Bamboogle	HotpotQA
Knowledge Conflict			
Vanilla LLM	100.0	100.0	100.0
Vanilla RAG	61.1	90.9	82.2
DeepNote	64.5	58.2	82.2
RAT	72.8	**92.7**	84.9
Search-o1	56.3	52.7	82.6
PAGER	**73.4**	83.6	**87.6**
Knowledge Utilization			
Vanilla LLM	0.0	0.0	0.0
Vanilla RAG	25.1	21.4	27.2
DeepNote	39.2	24.3	34.9
RAT	32.0	25.7	27.2
Search-o1	42.7	18.6	33.5
PAGER	**49.9**	**45.7**	**35.8**

Table 3: Performance of Different RAG Models under Different Testing Scenarios.

page initialization module.

5.4 The Impact of Different Representations on Knowledge Utilization in LLMs

In this section, we investigate how different knowledge representations affect knowledge utilization in LLMs. We adopt Qwen3-32B as the backbone model for all experiments.

As shown in Figure 5, we evaluate the effectiveness of constructed knowledge representations across different RAG modules. First, we analyze the document overlap between successive retrieval steps using the Jaccard similarity. As illustrated in Figure 5(a), the results show that PAGER exhibits substantially lower document overlap than DeepNote and RAT. This suggests that the page-based representation serves as an effective format for reducing retrieval redundancy across retrieval steps, thereby encouraging the model to incorporate more diverse knowledge instead of repeatedly retrieving similar information. Following prior work (Wang et al., 2025b), we further compute the information gain of different knowledge representations, as reported in Figure 5(b). Information gain quantifies the contribution of a knowledge representation to correct answer generation, with details provided in Appendix A.3. The results demonstrate that the pages constructed by PAGER achieve higher knowledge information gain than other methods. This indicates that the information encoded in structured page representations can better guide LLMs toward generating accurate answers.

Next, we evaluate the impact of different knowledge representations on knowledge conflicts and knowledge utilization in LLMs. As shown in Table 3, we design two evaluation scenarios: knowledge conflict and knowledge utilization. For the knowledge conflict scenario, we select samples from the evaluation dataset for which the LLM can generate correct answers solely based on its parametric knowledge, aiming to assess the denoising capability of RAG systems. For the knowledge utilization scenario, we construct the evaluation set by selecting samples on the LLM's answers must rely on external knowledge. The evaluation results show that both PAGER and RAT perform more effectively in the knowledge conflict scenario, indicating that incorporating the raw reasoning of LLMs to organize retrieved knowledge can alleviate conflicts between internal and external knowledge. In contrast, DeepNote and Search-o1 outperform RAT in the knowledge utilization setting, suggesting that incorporating more retrieved knowledge provides greater potential for correcting factual errors in memorized knowledge. Notably, PAGER achieves the best performance in the knowledge utilization scenario while maintaining performance comparable to RAT under the knowledge conflict setting. These results demonstrate that the constructed page representation offers a more tailored balance between mitigating knowledge conflicts and enhancing knowledge utilization.

6 Conclusion

This paper presents PAGER, a page-driven autonomous knowledge representation framework for RAG. By constructing a structured cognitive outline and iteratively filling it with retrieved evidence, PAGER organizes external knowledge into the structed page representation that better supports LLM to answer the questions. Experimental results show that PAGER consistently outperforms baselines across multiple tasks and backbone models.

Limitations

Although PAGER achieves superior performance across multiple datasets, the page construction process introduces additional latency. Specifically, to ensure the logical coherence and completeness of the constructed pages, PAGER employs an iterative slot-filling mechanism. This iterative process inevitably incurs extra computational overhead, leading to increased inference latency. Moreover, we also explore a variant of PAGER with parallel filling to mitigate this issue. However, experimental results indicate that the iterative design is still necessary to maintain the effectiveness of PAGER when answering multi-hop QA queries. Therefore, the trade-off between effectiveness and efficiency remains a critical challenge for fully deploying PAGER in real-world QA scenarios.

References

J. S. Bruner. 1966. Toward a theory of instruction. *Harvard University Press*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *ArXiv preprint*.

Kai Guo, Xinnan Dai, Shenglai Zeng, Harry Shomer, Haoyu Han, Yu Wang, and Jiliang Tang. 2025. Beyond static retrieval: Opportunities and pitfalls of iterative retrieval in graphrag. *ArXiv preprint*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of ICML*, pages 3929–3938.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2021. Efficient nearest neighbor language models. In *Proceedings of EMNLP*, pages 5703–5714.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of COLING*, pages 6609–6625.

Nan Huo, Jinyang Li, Bowen Qin, Ge Qu, Xiaolong Li, Xiaodong Li, Chenhao Ma, and Reynold Cheng. 2025. Micro-act: Mitigate knowledge conflict in question answering via actionable self-reasoning. *ArXiv preprint*.

Jinhao Jiang, Jiayi Chen, Junyi Li, Ruiyang Ren, Shijie Wang, Xin Zhao, Yang Song, and Tao Zhang. 2025. Rag-star: Enhancing deliberative reasoning with retrieval augmented verification and refinement. In *Proceedings of NAACL*, pages 7064–7074.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of EMNLP*, pages 7969–7992.

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

Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, and Ji-Rong Wen. 2025b. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 737–740.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, (3):535–547.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, pages 452–466.

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 SOSP*, pages 611–626.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of NeurIPS*.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025a. Search-o1: Agentic search-enhanced large reasoning models. *ArXiv preprint*.

Zhuoqun Li, Xuanang Chen, Haiyang Yu, Hongyu Lin, Yaojie Lu, Qiaoyu Tang, Fei Huang, Xianpei Han, Le Sun, and Yongbin Li. 2025b. Structrag: Boosting knowledge intensive reasoning of llms via inference-time hybrid information structurization. In *Proceedings of ICLR*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of EMNLP*, pages 5783–5797.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Proceedings of EMNLP Findings*, pages 5687–5711.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Proceedings of EMNLP Findings*, pages 9248–9274.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *ArXiv preprint*.

Zhongxiang Sun, Qipeng Wang, Weijie Yu, Xiaoxue Zang, Kai Zheng, Jun Xu, Xiao Zhang, Song Yang, and Han Li. 2025. Rearter: Retrieval-augmented reasoning with trustworthy process rewarding. *ArXiv preprint*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, pages 539–554.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of ACL*, pages 10014–10037.

Jesse Vig, Alexander Richard Fabbri, Wojciech Kryściński, Chien-Sheng Wu, and Wenhao Liu. 2022. Exploring neural models for query-focused summarization. In *Proceedings of NAACL*, pages 1455–1468.

Ruobing Wang, Qingfei Zhao, Yukun Yan, Daren Zha, Yuxuan Chen, Shi Yu, Zhenghao Liu, Yixuan Wang, Shuo Wang, Xu Han, Zhiyuan Liu, and Maosong Sun. 2025a. Deepnote: Note-centric deep retrieval-augmented generation. In *Proceedings of EMNLP*.

Zihan Wang, Zihan Liang, Zhou Shao, Yufei Ma, Huangyu Dai, Ben Chen, Lingtao Mao, Chenyi Lei, Yuqing Ding, and Han Li. 2025b. Infogain-rag: Boosting retrieval-augmented generation via document information gain-based reranking and filtering. *ArXiv preprint*.

Zihao Wang, Anji Liu, Haowei Lin, Jiaqi Li, Xiaojian Ma, and Yitao Liang. 2024. Rat: Retrieval augmented thoughts elicit context-aware reasoning in long-horizon generation. *ArXiv preprint*.

Mingyan Wu, Zhenghao Liu, Yukun Yan, Xinze Li, Shi Yu, Zheni Zeng, Yu Gu, and Ge Yu. 2025. Rankcot: Refining knowledge for retrieval-augmented generation through ranking chain-of-thoughts. In *Proceedings of ACL*.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2023a. Recomp: Improving retrieval-augmented lms with compression and selective augmentation. *ArXiv preprint*.

Ruochen Xu, Song Wang, Yang Liu, Shuohang Wang, Yichong Xu, Dan Iter, Pengcheng He, Chenguang Zhu, and Michael Zeng. 2023b. Lmgqs: A large-scale dataset for query-focused summarization. In *Proceedings of EMNLP Findings*, pages 14764–14776.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *ArXiv preprint*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of EMNLP*, pages 2369–2380.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *ArXiv preprint*.

Xiangrong Zhu, Yuexiang Xie, Yi Liu, Yaliang Li, and Wei Hu. 2025. Knowledge graph-guided retrieval augmented generation. In *Proceedings of NAACL*, pages 8912–8924.

A Appendix

A.1 License

We show the licenses of the datasets that we use. All of these datasets are allowed for academic use under their respective licenses and agreements: MuSiQue and HotpotQA (CC-BY-4.0 License); 2WikiMQA (Apache 2.0 License); Bamboogle (MIT License); NQ and AmbigQA (CC BY-SA 3.0 License).

A.2 More Implementation Details

In our experiments, we utilize FlashRAG (Jin et al., 2025b) to reproduce the results of IRCoT and Iter-RetGen on the experimental datasets. For RAT, Search-o1, StructRAG, and DeepNote, we reproduce their results using the official GitHub code provided. During inference, we use vLLM to load model checkpoints and perform offline batched generation, while setting unified hyperparameters: temperature = 0.7, top-p = 0.8, top-k = 20, and random seed = 66.

A.3 The Computation of Document Information Gain

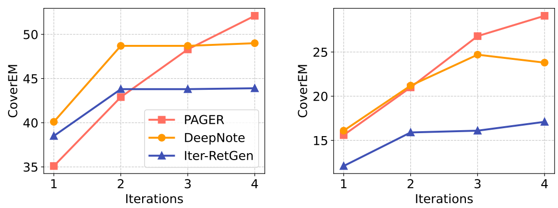
The document information gain (DIG) metric serves to quantify the actual utility of external knowledge within RAG systems (Wang et al., 2025b).

Formally, for a question q, a knowledge representation O, and the ground truth y, DIG is defined as the difference in the LLM's generation confidence for the correct answer when the K is included versus when it is excluded. Let $p_\phi(y \mid q, O)$ denote the conditional probability of the model generating the ground truth y when knowledge representation O is augmented into the context, and let $p_\phi(y \mid x)$ represent the probability of generating the ground truth y without the augmentation of knowledge representation O. Then, the DIG is calculated as:

$$\mathrm{DIG}(O \mid q) = p_\phi(y \mid q, O) - p_\phi(y \mid q), \quad (12)$$

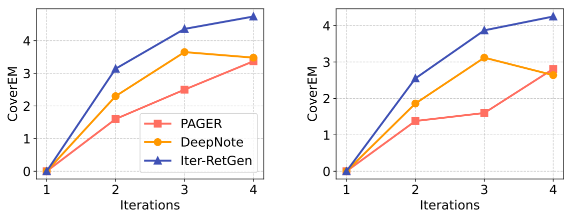
where ϕ denotes the parameters of the LLM. Furthermore, in the calculation of DIG, previous work introduces two strategies specifically designed to mitigate length bias and ensure the capture of the strongest signals indicative of generation quality:

Sliding Window Smoothing. To mitigate length bias in long sequences, a sliding window mechanism is utilized to smooth local probability fluctuations. For each token t_i in the answer sequence y,



(a) HotpotQA. (b) MuSiQue.

Figure 6: The Performance of Models Evolution across Iteration Rounds.



(a) HotpotQA. (b) MuSiQue.

Figure 7: The Document Overlap of Models Evolution across Iteration Rounds.

its smoothed probability is calculated as:

$$p_s(t_i) = \frac{1}{W} \sum_{j=i-\lfloor W/2 \rfloor}^{i+\lfloor W/2 \rfloor} p(t_j), \quad (13)$$

where W denotes the window size and $p(t_j)$ represents the original token probability.

Token Importance Weighting. To emphasize the core semantic information often encoded in initial tokens, a weighting scheme assigns higher weights to the first k tokens. The final calibrated probability score is derived as:

$$p_\phi(y|x) = \prod_{i=1}^{k} (p_s(t_i))^{\omega_i \cdot \alpha} \cdot \prod_{j=k+1}^{|y|} (p_s(t_j))^{1-\alpha}, \quad (14)$$

where ω_i is the importance weight for the i-th token, and α is a hyperparameter controlling the emphasis on the initial segment.

A.4 The Characteristics of Knowledge Representation Construction Processes

In this section, we further explore the effects of the retrieval and generation modules of different methods during the iterative construction of knowledge representations. We adopt Qwen3-32B as the backbone model for all experiments. We select the questions that require four rounds of iterative

Methods	HotpotQA	2WikiMQA	MuSiQue	Bamboogle	NQ	AmbigQA	Avg.
Vanilla LLM	39.0	57.0	15.5	62.4	58.0	53.0	47.5
Vanilla RAG	51.0	63.0	20.0	62.4	69.5	62.5	54.7
DeepNote (2025a)	45.0	41.5	20.5	52.0	52.0	55.0	44.3
PAGER	54.0	61.5	31.5	68.8	62.0	65.5	57.2

Table 4: Overall Performance of Different RAG Models. The **best** and <u>second best</u> results are highlighted. In our experiments, we employ GLM-4.5 as a backbone mode.

retrieval for PAGER, DeepNote, and Iter-RetGen to construct knowledge representations as the evaluation set.

As illustrated in Figure 6, we first analyze how the performance of each method evolves throughout the iterative process. We observe that, as the iteration count increases, the performance gains for Iter-RetGen and DeepNote rapidly plateau after the second round. This indicates that they encounter a bottleneck in continuously integrating new knowledge across multiple iterations, making it difficult to further capture and refine the key information in subsequent rounds. In contrast, PAGER exhibits steady performance improvement as iterations progress. This indicates that PAGER is capable of progressively incorporating more comprehensive external knowledge during the construction of structured page representations, thereby consistently enhancing model performance.

To better understand this behavior, we analyze the retrieval diversity in Figure 7, which displays the overlap between documents retrieved in the current iteration and the cumulative set of documents from all prior iterations. The results demonstrate that DeepNote and Iter-RetGen exhibit a higher degree of document overlap compared to PAGER. This high redundancy explains the performance plateau observed earlier. Conversely, the lower overlap observed in PAGER indicates its ability to retrieve more diverse knowledge and background information throughout the iterative process, ensuring the construction of a more comprehensive and robust knowledge representation.

A.5 The Performance of PAGER on Closed-source LLM

As shown in Table 4, we compare the overall performance of PAGER with various baseline methods on a range of knowledge-intensive tasks using GLM-4.5 as the backbone model. We sample 200 data points from each dataset for evaluation. Overall, when using GLM-4.5 as the backbone model, PAGER achieves performance improvements of over 3% compared to all baseline models, demonstrating its effectiveness. This indicates that PAGER is not only applicable to open-source LLMs such as Qwen3-32B and Llama3.1-70B-Instruct, but also demonstrates strong performance on closed-source LLMs.

A.6 Inference Time Latency

In this section, we compare the inference latency of PAGER (Parallel Filling) and PAGER in the ablation experiments presented in section 5.2. PAGER (Parallel Filling) simultaneously generates subqueries for all missing slots in the initial page and performs parallel retrieval. We utilize the Qwen3-32B as the backbone model.

As shown in Table 5, PAGER exhibits an inference latency that is approximately 1.8× that of PAGER (Parallel Filling). However, it is worth noting that PAGER typically operates with an average of four iteration rounds, indicating that the time cost does not grow linearly with the number of iterations (i.e., four iterations do not result in 4× latency). This is mainly because PAGER (Parallel Filling) needs to incorporate a larger number of retrieved documents from a single retrieval step into the context. More importantly, this additional temporal cost is justified by the performance gains, as the sequential iterations allow the model to dynamically adjust its retrieval strategy based on intermediate evidence, a capability that the parallel approach lacks.

A.7 Statistics of Page Slots

In this section, we further investigate the number of slots initialized by PAGER during page construction and their distribution. As shown in Figure 6, the number of slots in pages constructed by different backbone models is predominantly concentrated in the range of 3 to 5. This indicates that during page initialization, PAGER leverages its logical planning capability to focus the page structure on a set of key core topics. Further analysis reveals that for Qwen-32B, the pages it constructs predominantly contain four slots, whereas for Llama-72B, the constructed pages are mainly characterized by

Methods	HotpotQA	2WikiMQA	MuSiQue	Bamboogle	NQ	AmbigQA	Avg.
Inference Time Latency							
PAGER (Parallel Filling)	2.54	1.69	2.14	1.72	2.60	2.39	2.18
PAGER	3.82	3.42	3.88	3.76	4.38	3.71	3.82
Inference Performance							
PAGER (Parallel Filling)	45.9	43.8	18.8	59.2	56.4	57.0	46.9
PAGER	50.6	57.4	23.0	62.4	56.5	56.4	51.1

Table 5: Inference Time Latency. The unit of inference latency is seconds.

# Slots	HotpotQA	2WikiMQA	MuSiQue	Bamboogle	NQ	AmbigQA	Avg.
Qwen3-32B							
≤ 2	0.4	1.0	0.4	1.6	0.1	0.0	0.6
= 3	20.8	27.4	20.8	29.6	5.8	9.4	18.9
= 4	35.9	42.6	36.4	40.0	30.5	36.5	37.0
= 5	32.4	23.0	30.1	21.6	44.3	40.3	31.9
= 6	7.5	4.5	8.1	4.8	13.6	9.9	8.1
≥ 7	3.0	1.5	4.2	2.4	5.7	3.9	3.5
Llama3.1-70B-Instruct							
≤ 2	2.5	4.0	4.1	3.2	1.0	1.1	2.7
= 3	44.9	53.0	45.1	52.8	38.6	47.9	47.1
= 4	40.6	37.3	37.2	34.4	36.1	35.8	36.9
= 5	10.4	5.6	11.4	9.6	19.4	13.0	11.6
= 6	1.4	0.1	1.9	0.0	3.8	1.8	1.4
≥ 7	0.2	0.0	0.3	0.0	1.1	0.4	0.3

Table 6: The Statistics of Page Slots.

three slots. This phenomenon further indicates that, due to the varying inherent logical reasoning and cognitive capabilities of different LLMs, the structure of the initialized cognitive outlines and the distribution of slot counts also differ.

A.8 Case Studies of PAGER

In this section, we select some cases to demonstrate the effectiveness of PAGER, as well as the processes of page construction of PAGER. All cases are selected from the HotpotQA dataset.

As illustrated in Figure 8, we compare the performance of three models: Vanilla RAG, DeepNote, and PAGER. Vanilla RAG and DeepNote erroneously identify "Bill Nye" and "Wil Wheaton", respectively, as the correct actors, despite neither having starred in "The Bronze". In contrast, the page constructed by PAGER accurately consolidates the key information that "Melissa Rauch" starred in "The Bronze" and also appeared in The Big Bang Theory. This demonstrates that by constructing a comprehensive knowledge representation, PAGER effectively assists the LLM in answering the question.

Figure 9 further illustrates the iterative page construction process of PAGER. During the initialization phase, PAGER constructs a structured page outline tailored to the query, establishing designated slots. This structural outline serves as a roadmap to guide the subsequent knowledge acquisition and utilization. In Iterations 1 and 2, the PAGER generates sub-queries to retrieve external documents, successfully refining key information into the page. Subsequently, in Iterations 3 and 4, the model formulates sub-queries designed to explore the relationship between the entities. Retrieval results reveal that the two directly competed in a 2000 tournament. Consequently, PAGER synthesizes this evidence within the "Career Comparison", explicitly articulating that both subjects are deeply involved in the realm of MMA. Finally, leveraging this comprehensively constructed and logically coherent page, the model not only identifies the correct answer, "Mixed martial artist", but also grounds its conclusion by citing specific evidence embedded within the page.

A.9 Prompt Templates Used for PAGER

We provide a detailed description of the instruction prompts used in the experiments of PAGER. Figure 10 illustrates the instruction for initializing the outline of the page. Figure 11 shows the instructions for generating sub-queries. Figure 12 presents the instruction used for filling the page slots. Figure 13 shows the instructions for evaluating the overall quality of the knowledge representations generated by different models.

Case #1: Effectiveness between Different Model

Question: Who starred in The Bronze and also showed up on the CBS sitcom "The Big Bang Theory"?

Vanilla RAG

[id: 1]: …Hawking video conferences with Sheldon and Leonard, and makes another appearance in the 200th episode. In the fifth and sixth seasons, NASA astronaut Michael J. Massimino played himself multiple times in the role of Howard\'s fellow astronaut. ***Bill Nye appeared in the seventh and twelfth seasons.***

[id: 2]: ***The Big Bang Theory The Big Bang Theory is an American television sitcom created by Chuck Lorre and Bill Prady, both of whom serve as executive producers on the series, along with Steven Molaro.*** All three also serve as head writers. The show premiered on CBS on September 24, 2007…

…

[id: 4]: The third season saw the first appearances of future main cast members ***Melissa Rauch*** as Bernadette Rostenkowski in ""The Creepy Candy Coating Corollary"…

Final Answer: Bill Nye (Incorrect)

DeepNote

These include theoretical physicist Brian Greene, astrophysicist Neil deGrasse Tyson, cosmologist Stephen Hawking, NASA astronaut Michael J. Massimino, and Bill Nye, among others. No individual from this list is explicitly identified as having starred in *The Bronze*. Wil Wheaton, however, is mentioned as having appeared as a fictionalized version of himself in several episodes of *The Big Bang Theory*, starting in season 3, episode 5: *The Creepy Candy Coating Corollary* (2009). ***On the show, Wheaton behaves in comically petty and manipulative ways towards main character Sheldon Cooper, who regards him as an archenemy until the season 5 episode *The Russian Rocket Reaction*, when they make amends and become friends.*** Wheaton appears in 12 episodes in a recurring, guest-starring role on *Eureka*, playing Dr. Isaac Parrish, the head. Wil Wheaton also hosts *TableTop*, a board game based show, and *Titansgrave*, a roleplaying game based show.While the retrieved documents do not explicitly mention Wil Wheaton's role in *The Bronze*, it is known from external sources that Wil Wheaton starred in the 2016 comedy film *The Bronze*. ***Therefore, Wil Wheaton is the individual who starred in *The Bronze* and also appeared on *The Big Bang Theory*, as detailed in the retrieved documents.***

Final Answer: Wil Wheaton (Incorrect)

PAGER

Who Starred in The Bronze and Also Appeared on The Big Bang Theory?

Overview of The Bronze (2014) \n*The Bronze* is a 2015 American sports comedy-drama film directed by Bryan Buckley and written by Melissa Rauch and Winston Rauch. The film centers around Hope Ann Greggory, a former gymnast who won a bronze medal in the 1996 Olympics and is struggling to maintain relevance in the spotlight. It was produced by Mark Duplass and Jay Duplass through their Duplass Brothers Productions…

Notable Cast Members of The Bronze \nThe film *The Bronze* features a notable ensemble cast led by Melissa Rauch, who also co-wrote the screenplay and portrays the lead character, Hope Ann Greggory. Other prominent cast members include Gary Cole, Thomas Middleditch, Sebastian Stan, Cecily Strong, Haley Lu Richardson, and Dale Raoul. Rauch's role in the film is particularly significant, as she brings a comedic and self-deprecating energy to the character of a former Olympic gymnast trying to reclaim her relevance…

…

Identifying the Actor Common to Both Projects

The actor who starred in *The Bronze* and also appeared on *The Big Bang Theory* is Melissa Rauch. In *The Bronze*, Rauch played the lead role of Hope Ann Greggory, a disgraced Olympic gymnast navigating the challenges of public life…

Final Answer: Melissa Rauch (Correct)

Figure 8: Case Study of Different Methods. Text in **red** represents document excerpts directly related to the knowledge representation and the question answering, while text in **blue** represents the noisy content in the knowledge representation.

Case #2: Page Construction Process of PAGER

Question: What profession does Kazuyuki Fujita and Gilbert Yvel have in common?
Ground Truth: Mixed martial artist

Init Page:
Shared Profession of Kazuyuki Fujita and Gilbert Yvel
Introduction to Kazuyuki Fujita
<TO BE FILLED>
Introduction to Gilbert Yvel
<TO BE FILLED>
Career Comparison: Kazuyuki Fujita vs. Gilbert Yvel
<TO BE FILLED>
Common Profession: Identifying the Overlap
<TO BE FILLED>

Generate Sub-query:
What is Kazuyuki Fujita's profession?
Retrieved Documents:
Kazuyuki Fujita Fujita practiced freestyle wrestling in high school. Championships as a junior in 1988, placing sixth, and in the Espoir division in 1989He participated in the FILA World, placing eleventh. In 1993, Fujita placed fifth at the Asian Championships at the senior level, and in 1993 and 1994, he represented Japan as a senior in the World Cup...

Fill Page (iter-1):
Shared Profession of Kazuyuki Fujita and Gilbert Yvel
Introduction to Kazuyuki Fujita
Kazuyuki Fujita is a Japanese professional wrestler and mixed martial artist who began his career in freestyle and Greco-Roman wrestling during his high school and early adult years. He competed in several...

...

Generate Sub-query:
What is Gilbert Yvel's profession?
Retrieved Documents:
Gilbert Yvel Gilbert Ramon Yvel (born June 30, 1976) is a Dutch mixed martial artist currently competing in the Super Heavyweight division and associated with the Vos Gym in the Netherlands. A professional competitor for 16 years from 1997-2013,...

Fill Page (iter-2):

...

Introduction to Gilbert Yvel
Gilbert Yvel, born Gilbert Ramon Yvel on June 30, 1976, is a Dutch mixed martial artist best known for his aggressive fighting style and controversial behavior inside the cage. Competing primarily in the super heavyweight division, Yvel has participated in numerous high-profile promotions...

...

Generate Sub-query:
What are the similarities in the careers of Kazuyuki Fujita and Gilbert Yvel in combat sports?
Retrieved Documents:
Cold Fury show in 2000, in the 7th fight of the evening, former Japan national Greco-Roman wrestling champion Kazuyuki Fujita faced Dutch Muay Thai stylist Gilbert Yvel, in a classic wrestler versus striker mixed martial arts (MMA) matchup...

Fill Page (iter-3):

...

Career Comparison: Kazuyuki Fujita vs. Gilbert Yvel
Both Kazuyuki Fujita and Gilbert Yvel share a deep involvement in combat sports, particularly within the realm of mixed martial arts (MMA). Fujita, a former Greco-Roman wrestling champion, transitioned to MMA after a...

...

Generate Sub-query:
What profession do Kazuyuki Fujita and Gilbert Yvel share?
Retrieved Documents:
a commentator with a ""foreign"" point of view. He is known as one of ""the two Kents"" (along with Kent Derricott). Gilbert lives in Tokyo and also maintains a residence in Orem, Utah. He made an...

Fill Page (iter-4):

...

Career Comparison: Kazuyuki Fujita vs. Gilbert Yvel
Both Kazuyuki Fujita and Gilbert Yvel share a deep involvement in combat sports, particularly within the realm of mixed martial arts (MMA). Fujita, a former Greco-Roman wrestling champion, transitioned to MMA after a...

...

Answer Generation:
...including their notable bout in the PRIDE Fighting Championships. The section titled "Common Profession: Identifying the Overlap" directly addresses the question and confirms their shared profession.
<answer> Mixed martial artist </answer>

Figure 9: Case Study of Iterative Page Construction. Text in **red** represents the knowledge evidence filled into the slots.

Instruction for Outline Initialization

You are a Page Planning Expert.
For a given question, you are tasked with generating a page outline based on the theme of the question.
The outline you generate will serve as the foundation for the subsequent process, during which the outline will be filled with content to create a complete page that assists the reader in answering the given question.
Please strictly follow the instructions below.

Input:
- Question: {question}

Task Steps:

1. **Reasoning Analysis**
- First, analyze the theme of the question and thoroughly understand the knowledge required to answer it, as well as the logical relationships between this knowledge.
- Based on the theme of the question, generate a page outline and identify the key content that should be covered in each section.
- For each section, propose a suitable and appropriate title, ensuring that each title effectively guides the reader to progressively deepen their understanding of the various aspects of the question.

1. **Outline Initialization**
- Generate the outline:
- Use # [Main Title] for the main title. The main title is a concise and comprehensive abstraction of the page content.
- Use ## [Section Title] for each section title. The section title is the section heading (do not reveal the final answer).
- Insert special marker <TO BE FILLED> under all section titles.
Make sure the sections of the page are ordered logically, building up the reader's understanding toward answering the question.
The number of sections on the page should be limited to the scope necessary to answer the question, avoiding any overlap of content between sections.

2. **Output**
- First, you should output the reasoning analysis for initializing the outline.
- Then, you should generate a special symbol <OUTLINE>, followed by the generated page outline. Note that the content after <OUTLINE> should only include the final generated page outline, without any comments or explanations.

Figure 10: The Prompt Template for Generating the Outline.

Instruction for Sub-query Generation

You are a professional question-generation expert.
First, following the order of the page sections, locate and identify the first unfinished section. An unfinished section is defined as one that contains the special symbol <TO BE FILLED> under the section title.
Please strictly follow the steps and format below to formulate a precise retrieval question for the first unfinished section of the current page, in order to obtain the necessary information to complete that section.

Input Parameters:
- Original question: {question}
- Current page content: {page}

Task Steps:

1. **Parse the page plan:**
- Read the Page and locate the title and topic of the first section that remains unfilled.

2. **Align with the original question:**
- Align the section's topic with the core needs of original question to ensure the retrieval question directly serves the original question.

3. **Design the retrieval question. The question must:**
- Be focused: target only the core subtopic of that section;
- Be clear: use search terms that can be directly used in a search engine or knowledge base;
- Be concise: include no unnecessary background.

4. **Output format:**
- Output only one line containing the English retrieval question, with no additional comments or explanations.

Figure 11: The Prompt Template for Generating the Subquery.

Instruction for Page Filling

You are a professional page content writer.
Please strictly follow the instructions below.
First, following the order of the page sections, locate and identify the first unfinished section. An unfinished section is defined as one that contains the special symbol <TO BE FILLED> under the section title.
Use the retrieved documents, and your internal knowledge to complete the first unfinished section of the page, and generate the page with that section filled in.
Please note that you should only fill in one unfinished section, and that section must be the first unfinished section on the page. Do not fill in additional sections or fill across different sections.

Input:
- Original question: {question}
- Sub-question for retrieval: {sub_question}
- Retrieved documents: {docs_text}
- Current page: {page}

Task Steps:

1. **Section Completion**
- Find the first unfinished section in the page, where the section title contains a special placeholder <TO BE FILLED>.
- Using information from the retrieved documents and your internal knowledge, write a short paragraph under that section. The paragraph should:
- Be tightly related to the original question.
- Focus strictly on the topic of that section.
- Avoid redundant or irrelevant information.
- Remove the <TO BE FILLED> placeholder under that section.
- Retain <TO BE FILLED> placeholders for all other unfinishe sections.
- If you have filled the last unfinished section of the page, ensure that no <TO BE FILLED> placeholders remain in the page.
2. **Output format**
- Only output the entire page with the first unfinished section fully filled in. Do not include any comments, explanations, or isolated section content.
- Your output must be the entire updated page, including the newly filled content, seamlessly integrated into the original page structure.
- Do not output just the section you filled in—your output must be the entire page, including all content, both existing and newly added.

Figure 12: The Prompt Template for Filling the Slots in the Page.

The Instruction for Using GLM to Evaluate Knowledge Representations

You are an expert evaluator proficient in cognitive linguistics and natural language processing tasks. Your task is to evaluate the quality of a "Knowledge Context". This context was generated by a model based on raw retrieved documents following multi-round knowledge retrieval and refinement, specifically for the purpose of answering a given question.

Input Data
- **User Question**: {question}
- **Ground Truth Answer**: {ground_truth}
- **Raw Retrieved Documents**: {retrieved_docs}
- **Evaluated Knowledge Context**: {generated_context}

Evaluation Criteria
Please strictly review the "Evaluated Knowledge Context" based on the following four core dimensions. **You must assign a separate score (0-5) for each dimension.**

1. **Knowledge Structuring**
- Does the Knowledge Context establish a coherent knowledge framework or outline, rather than merely listing scattered facts?
- Are key concepts hierarchically organized or classified to facilitate the understanding of complex information?
- *Negative Indicators*: Fragmented information, lack of a main thread, keyword stuffing.

2. **Document Refinement**
- Does the Knowledge Context demonstrate the "digestion" and "reorganization" of the original documents? Does it bridge logical gaps by synthesizing information from multiple documents?
- Has the raw material been transformed into a problem-oriented explanation, rather than simple copy-pasting?
- *Negative Indicators*: Mechanical repetition, concatenation of excerpts, failure to adapt content to the specific question.

3. **Coherence & Logical Flow**
- Is the narrative of the context fluid? Are there clear causal or progressive relationships between ideas?
- Does it resolve potential contradictions found in the raw documents or reasonably reconcile conflicting information?
- *Negative Indicators*: Logical jumps, self-contradiction, abrupt transitions.

4. **Correctness & Argumentation**
- Is the evidence provided in the Knowledge Context sufficient and accurate enough to forcefully support the derivation of the "Standard Answer"?
- *Negative Indicators*: Missing key evidence, inclusion of misleading information, irrelevance to the Standard Answer.

Scoring Scale (Applied to EACH dimension independently)
Please evaluate each dimension on a scale of 0 to 5 using the rubric below:

- **5 (Perfect)**: The dimension is executed flawlessly. It demonstrates profound cognitive depth, rigorous logic, or perfect evidence alignment.
- **4 (Excellent)**: Strong execution. Meets the high standards of the dimension with only negligible flaws.
- **3 (Satisfactory)**: Acceptable performance. Contains the main necessary elements but lacks depth or relies on basic stitching rather than deep processing.
- **2 (Poor)**: Weak performance. Significant gaps exist in this specific dimension (e.g., chaotic structure, poor synthesis, or logical breaks).
- **1 (Unusable)**: Severe failure. The context fails almost completely in this dimension (e.g., hallucinations, complete lack of logic, or missing critical evidence).
- **0 (Failure)**: Completely irrelevant, empty, or non-existent effort in this dimension.

Output Format
Please strictly follow the format below to present your evaluation. Do not output conversational filler.

1. Knowledge Structuring
- **Score**: [0-5]
- **Reasoning**: [Critically analyze the framework and hierarchy. Cite specific examples of good or bad structuring.]

2. Document Refinement
- **Score**: [0-5]
- **Reasoning**: [Evaluate the extent of information digestion and reorganization. Did it simply copy-paste or truly synthesize?]

3. Coherence & Logical Flow Evaluation
- **Score**: [0-5]
- **Reasoning**: [Analyze the narrative fluidity and causal connections between sentences/paragraphs.]

4. Correctness & Argumentation Evaluation
- **Score**: [0-5]
- **Reasoning**: [Assess if the evidence provided is accurate and sufficient to support the Ground Truth.]

Final Summary
- **Average Score**: [Calculate the arithmetic mean of the 4 scores]
- **Conclusion**: [One sentence summary of the overall quality.]

Figure 13: The Prompt Template for Scoring Using the GLM Model.