HoH: A Dynamic Benchmark for Evaluating the Impact of Outdated Information on Retrieval-Augmented Generation

Jie Ouyang, Tingyue Pan, Mingyue Cheng*, Ruiran Yan, Yucong Luo, Jiaying Lin, Qi Liu
State Key Lab of Cognitive Intelligence, University of Science and Technology of China
{ouyang_jie,pty12345,yanruiran,prime666,linjya}@mail.ustc.edu.cn
{mycheng,qiliuql}@ustc.edu.cn

Abstract

While Retrieval-Augmented Generation (RAG) has emerged as an effective approach for addressing the knowledge outdating problem in Large Language Models (LLMs), it still faces a critical challenge: the prevalence of outdated information in knowledge bases. Current research primarily focuses on incorporating up-to-date information, yet the impact of outdated information coexisting in retrieval sources remains inadequately addressed. To bridge this gap, we introduce HoH, the first benchmark specifically designed to evaluate the impact of outdated information on RAG. Our benchmark leverages token-level *diff* algorithms combined with LLM pipelines to efficiently create a large-scale QA dataset that accurately captures the evolution of temporal knowledge in real-world facts. Through comprehensive experiments, we reveal that outdated information significantly degrades RAG performance in two critical ways: (1) it substantially reduces response accuracy by distracting models from correct information, and (2) it can mislead models into generating potentially harmful outputs, even when current information is available. Current RAG approaches struggle with both retrieval and generation aspects when handling outdated information. These findings highlight the urgent need for innovative solutions to address the temporal challenges in RAG. Our code and data are available at https://github.com/0russwest0/HoH.

1 Introduction

Large Language Models (LLMs) have revolutionized natural language processing with their remarkable knowledge retention and reasoning abilities (OpenAI et al., 2024; Grattafiori et al., 2024; Qwen, 2024). However, the static nature of their pretrained knowledge poses a significant challenge in handling rapidly evolving real-world information. LLMs frequently generate outdated responses

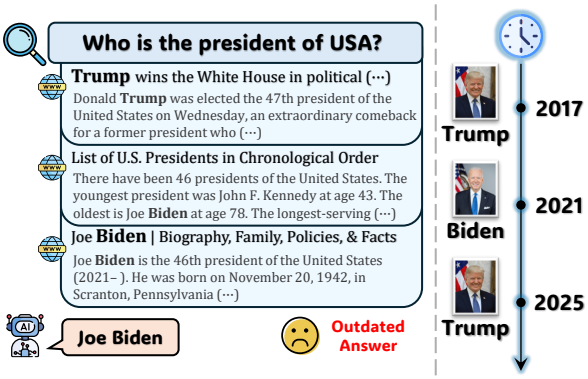

Figure 1: An illustration of how outdated information affects RAG. The example shows a query about the US president, where the retrieved passages contain outdated information, leading to potential confusion in generating an accurate response.

that, while appearing plausible, no longer reflect current facts (Maynez et al., 2020; Liu et al., 2023). Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Guu et al., 2020; Izacard and Grave, 2021; Borgeaud et al., 2022; Yu et al., 2024) has emerged as a promising solution to this knowledge-outdating problem by dynamically retrieving and integrating up-to-date information from external sources.

Despite RAG's promise, a critical challenge remains unaddressed: the pervasive presence of outdated information in knowledge bases (Xin et al., 2024). As illustrated in Figure 1, when querying about the current US president, RAG systems (Cheng et al., 2025; Ouyang et al., 2024) may retrieve both current and outdated information, potentially leading to confusion or incorrect responses. This scenario is commonplace as information evolves and outdated facts inevitably accumulate across various sources, particularly in search engine scenarios where content is cached and redistributed. Current research primarily focuses on how to effectively retrieve and leverage the latest information (Karpukhin et al., 2020; Chen et al., 2024), while overlooking how the presence of outdated information could harm RAG. This

* Corresponding author.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6036–6063

oversight is significant, as even state-of-the-art industry RAG systems (Mehdi, 2023) frequently generate incorrect responses due to outdated information (Vu et al., 2024). To address this challenge, we introduce **HoH** [1] (**H**ow **o**utdated information **H**arms Retrieval-Augmented Generation), the first large-scale benchmark designed to evaluate RAG's robustness against outdated information.

HoH comprises two primary components: HoH-QA and HoH-SearchEngine. HoH-QA is a continuously evolving QA dataset that maintains temporal alignment with contemporary world knowledge. Our dataset features comprehensive evidence annotations for each QA pair, enabling fine-grained evaluation of individual RAG components. More importantly, we introduce annotations for outdated answers and evidence for the first time, allowing systematic assessment of how outdated information impacts RAG performance. Complementing the QA dataset, HoH-SearchEngine simulates real-world search scenarios by maintaining both current and historical documents, better reflecting the actual challenges RAG faces in practice. Our benchmark currently includes 96,124 QA pairs and 219,463 documents, far surpassing the scale of existing datasets.

To ensure data quality and efficiency, we develop novel methods for dataset construction and maintenance. We propose a two-stage approach that combines *diff* algorithms (Myers, 1986) with LLM pipelines to extract and verify factual changes from Wikipedia snapshots. This approach significantly outperforms previous methods (Jang et al., 2022; Kim et al., 2024; Ko et al., 2024) in both efficiency and accuracy. The automated pipelines include rigorous quality control mechanisms, ensuring the reliability of the generated data while maintaining the dataset's continuous evolution.

Through extensive experiments on the HoH benchmark, we demonstrate that outdated information significantly degrades RAG performance in two critical ways: (1) outdated information substantially reduces response accuracy: even when current information is successfully retrieved, the mere presence of outdated information in the context leads to at least 20% performance drop in mainstream LLMs, with some models performing worse than random guessing (-2.77%); (2) outdated information can mislead models into generating harmful outputs: while models maintain appropriate uncertainty when no information is retrieved ("Unknown" or "Unsure"), they become highly prone to generating confident but incorrect responses when encountering outdated information. These findings highlight the urgent need for innovative solutions to address the temporal challenges in RAG.

In summary, our main contributions are:

1. We introduce HoH, the first large-scale dynamic QA benchmark for evaluating RAG's robustness against outdated information, featuring comprehensive evidence annotations and a mock search engine that maintains both current and historical documents.
2. We develop an efficient and scalable dataset construction method that combines traditional *diff* algorithms with LLMs, achieving superior quality in factual change extraction while maintaining continuous dataset evolution.
3. We present the first systematic analysis demonstrating the harmful impact of outdated information on RAG, providing crucial insights for future research in this domain.

2 Background and Related Work

HoH belongs to a growing body of works aimed at handling Time-Sensitive Questions in Question Answering (QA) systems. Time-Sensitive Questions are queries whose answers may vary depending on when the question is asked or the temporal context specified (Chen et al., 2021). Compared to traditional QA tasks (Dinan et al., 2018; Kwiatkowski et al., 2019), Time-Sensitive QA demands the effective utilization of specific temporal contexts, encompassing multiple time-evolving facts, to address time-sensitive questions (Yang et al., 2024a). Based on whether temporal contexts are explicitly specified, Time-Sensitive Questions can be categorized into two types: Time-Situated Questions and Present-Anchored Questions.

2.1 Time-Situated Question Answering

Early research primarily focused on Time-Situated Questions (Chen et al., 2021; Zhang and Choi, 2021; Liska et al., 2022; Dhingra et al., 2022), which contain explicit temporal contexts (e.g., "Who was the President of the USA in 2023?"). Subsequent works (Jia et al., 2018; Saxena et al., 2021; Tan et al., 2023) introduced more complex temporal reasoning tasks. These datasets not only contain questions with temporal contexts but also

[1] HoH resembles a shocked emoji 😱, expressing our alarm at the harm caused by outdated information.

Table 1: Comparison of our HoH with existing dynamic benchmarks. The automation indicates the feasibility of automatic generation. The Maintenance represents whether the validity of previously generated datasets is verified in the forthcoming time step. The evidence text indicates whether the dataset includes the evidence text. The outdated info shows whether the dataset contains annotation for outdate information. The 'question number' specifically counts Time-Sensitive Questions (excluding time-irrelevant ones), and 'article number' identifies the number of articles for retrieval, respectively.

	Automation	Maintenance	Evidence Text	Outdated Info	Question Number	Article Number
RealtimeQA (Kasai et al., 2023)	✗	✗	✗	✗	2,340	16,023
FreshQA (Vu et al., 2024)	✗	✗	✗	✗	600	-
TemporalWiki (Jang et al., 2022)	✓	✗	✗	✗	-	-
EvolvingQA (Kim et al., 2024)	✓	✗	✗	✗	23,283	-
GrowOVER (Ko et al., 2024)	✓	✓	✓	✗	1,257	-
PAT-Questions (Meem et al., 2024)	✓	✓	✗	✗	6,172	-
CLARK-News (Li et al., 2024)	✗	✗	✓	✓	1,409	1,149
HoH (Ours)	✓	✓	✓	✓	96,124	219,463

require temporal reasoning capabilities (e.g., "Who was the President of the USA before Obama?"). While these benchmarks have advanced our understanding of temporal reasoning in QA systems, they do not address the challenges posed by continuously evolving information and the need for up-to-date answers in real-world applications.

2.2 Present-Anchored Question Answering

In contrast to Time-Situated Questions, Present-Anchored Questions implicitly require the most current information available (e.g., "Who is the President of the USA?"). These questions pose unique challenges as their answers may change over time, necessitating continuous updates to maintain accuracy. Recent years have witnessed several attempts to address these challenges through dynamic benchmarks. Early approaches, such as RealtimeQA and FreshQA (Kasai et al., 2023; Vu et al., 2024), rely on manual curation to ensure high-quality QA pairs, but suffer from limited scalability and impractical update frequencies. To overcome these limitations, automated approaches have emerged. TemporalWiki and EvolvingQA (Jang et al., 2022; Kim et al., 2024) demonstrate the feasibility of automatic dataset generation, while GrowOVER and PAT-Questions (Ko et al., 2024; Meem et al., 2024) advance this further by incorporating maintenance mechanisms. Recently, CLARK-News (Li et al., 2024), in addition to providing the latest answers, is the first to record the historical progression of answer changes over time. However, due to its manual construction approach, it faces challenges in maintaining updates and is limited in scale.

As shown in Table 1, our work, HoH, builds upon these foundations while addressing key limitations in existing benchmarks. Through automation and continuous maintenance, HoH ensures both scalability and data validity. It includes evidence text annotations, enabling fine-grained analyses of both retrieval and generation components in RAG systems. More importantly, it uniquely provides annotations for outdated information, enabling systematic study of its impact. Lastly, HoH offers unprecedented scale with 96,124 questions and 219,463 articles, surpassing existing benchmarks in comprehensiveness.

3 The HoH Benchmark

HoH benchmark comprises two key components: HoH-QA, a dynamic QA dataset that tracks temporal changes in facts, and HoH-SearchEngine, a mock search engine that simulates real-world information retrieval scenarios. In this section, we detail our methodology for constructing and maintaining these components.

Article Selection. HoH is based on Wikipedia snapshots [2], which contain a vast amount of world knowledge. We select almost all articles from the snapshots, excluding only low-quality articles with excessively short content (less than 200 characters). The initial version was generated based on snapshots from 2024-06-01 and 2024-07-01, with regular monthly updates since then. As of now, we have updated it to the snapshot of 2024-11-01. For more detailed statistics, see Appendix C.

3.1 HoH-QA

HoH-QA is a dynamic open-domain QA dataset that tracks temporal changes in facts through two complementary mechanisms: (1) capturing time-sensitive questions whose ground-truth answers

[2]Download from https://dumps.wikimedia.org/

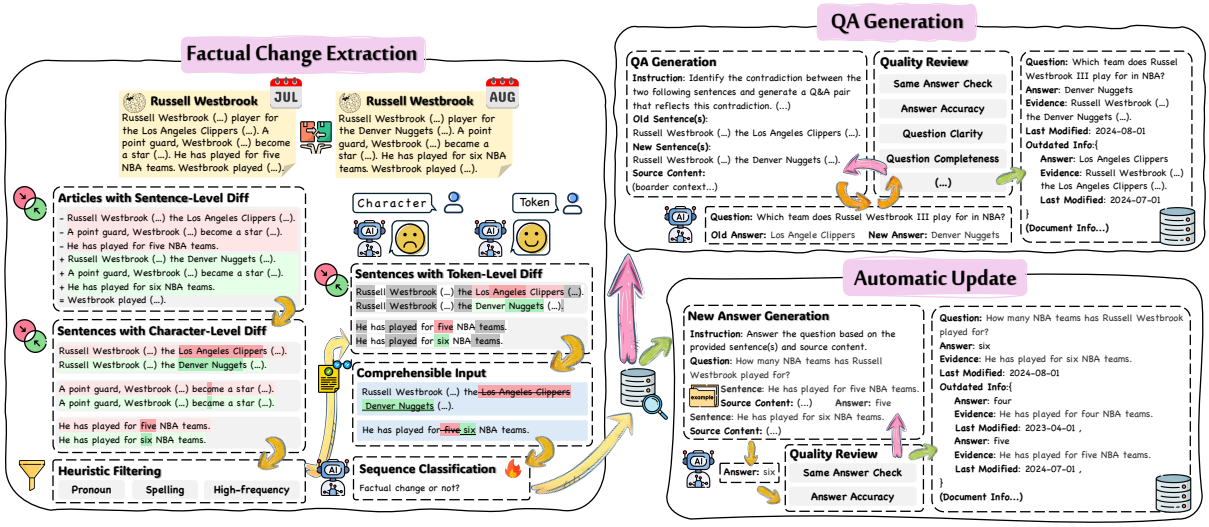

Figure 2: The overall process of HOH-QA construction comprises three primary parts: factual change extraction, QA generation, and automatic update. In the example shown, changes are identified between the July and August versions of a Wikipedia article. First, sentence pairs corresponding to factual changes are extracted. These extracted sentence pairs are then matched against previously generated datasets. For new sentence pairs, a QA generation process is conducted to create QA pairs. For sentence pairs that already exist, an automatic update process is employed to ensure the information remains current and accurate.

evolve over time, and (2) maintaining continuous dataset updates to reflect the latest world knowledge. Following Jang et al. (2022); Kim et al. (2024); Ko et al. (2024), we generate HOH-QA through comparing two different Wikipedia snapshots at different time points. Building upon their approach, we enhance both the efficiency and quality, and additionally incorporate annotations for outdated information.

As illustrated in Figure 2, the main process comprises three primary parts: factual change extraction, QA generation, and automatic update.

Factual Change Extraction. The initial phase of factual change extraction is to identify the modified sections across the same article of two different snapshots. We first split the old article A_{old} and new article A_{new} into sentences, get $A_{old} = \{s_{old_1}, s_{old_2}, \cdots, s_{old_n}\}$ and $A_{new} = \{s_{new_1}, s_{new_2}, \cdots, s_{new_m}\}$. Following Kim et al. (2024), we employ the Myers *Diff* [3] (Myers, 1986) algorithm to detect modified sentence pairs. *Diff* algorithms are commonly employed to compare differences between texts, code (e.g., in Git), and are generally applicable to any sequence. By treating an article as a sequence of sentences, we obtain modified sentence pairs.

While prior works (Jang et al., 2022; Kim et al., 2024) extract changes only at the sentence level, modified sentence pairs do not necessarily correspond to factual alterations. We further filtered these pairs and applied a *diff* algorithm at the character level to identify differences between sentences. For character-level differences, we employ heuristic methods to filter out modifications that are apparently not factual changes.

To further ensure the quality of extracted factual changes, we introduce a semantic screening stage using language models (LMs). Inspired by the human approach of reviewing different code versions, we consolidated the sentences before and after modification and delineated the differences between sentences in a comprehensible manner. For added parts, we use underlines for marking, and for deleted parts, we use strikethrough. To ensure that the marked differences align with the model's reading habits, we do not process differences at the character level. Instead, we combine the *diff* algorithm with the tokenizer, handling differences at the token level.

In building a reliable screening model, we constructed a high-quality training dataset where three annotators independently verified 2,000 sentence pairs from our heuristically filtered samples. The resulting fine-tuned model[4] achieves 96.8% accuracy and 95.1% F1 score in identifying genuine factual changes, outperforming previous methods by a significant margin. Complete details of the fine-tuning process and comparative evaluations

[3]Implemented by diff-match-patch

[4]Based on Qwen2.5-0.5B (Qwen, 2024)

are provided in Appendix B.2 and A.1.

QA Generation. Following established practices in automatic QA generation (Trischler et al., 2017; Rajpurkar et al., 2018), we employ LLMs to create question-answer pairs from the extracted factual changes. For each identified change, we generate a question that captures the temporal aspect, along with two distinct answer versions: the current correct answer and its outdated counterpart. The corresponding sentence pairs serve as evidence, labeled as current and outdated evidence accordingly.

The quality of generated QA pairs is ensured through a multi-stage review process. After initial generation, LLMs conduct a thorough review, focusing on the quality of both questions and answers. Detailed generation prompts, quality control procedures, and a manual validation confirming the high reliability of this automated process are provided in Appendix B.3.

Automatic Update. Our dataset maintains synchronization with evolving knowledge through monthly updates based on new Wikipedia snapshots. For each update cycle, we apply our factual change extraction pipeline to identify modifications between consecutive snapshots. The update process handles two scenarios: (1) newly emerged facts, where we directly apply our QA generation pipeline, and (2) previously captured facts that undergo further changes, which require special handling.

For facts with continuous changes, we leverage LLMs to generate updated answers to existing questions using the latest information. This approach preserves the question's temporal relevance while creating a chain of answers that reflects the evolution of facts over time. Similar to our QA generation process, we employ LLMs to review the quality of updated answers. After confirming their accuracy and distinctness from previous versions, these answers are incorporated into our dataset, with each superseded answer preserved and marked as outdated.

3.2 HoH-SearchEngine

As QA datasets evolve over time, the corresponding external knowledge sources for retrieval must maintain temporal consistency. Search engines, being one of the most common forms of external knowledge sources, offer convenience for knowledge updates. We implement HoH-SearchEngine using Elasticsearch[5] (Elasticsearch, 2018), extending its default BM25-based ranking (Robertson et al., 2009) with temporal awareness. Specifically, we introduce a Gaussian decay function to discount outdated information, mimicking the temporal preference of real-world search engines. The underlying corpus comprises both current and historical versions of all articles used in the QA dataset construction, simulating the prevalence of outdated information on the web.

[5] https://www.elastic.co/elasticsearch

4 Experiment Setup

In this section, we present definitions for document/passage categories, introduce time-aware instruction, explain the data and models tested in our experiments, and discuss the metrics and evaluation methods.

4.1 Type of Documents/Passages

In our study, we categorize documents/passages into three distinct types, each represented by a unique symbol, based on their relevance to the questions:

Relevant ($\mathcal{R}$): These passages contain the correct answer and are useful for the question.

Outdated ($\mathcal{O}$): These are previously relevant but have become obsolete, no longer accurate due to temporal changes.

Distracting ($\mathcal{D}$): These documents are semantically similar to the question but do not contribute correct answers.

4.2 Time-Aware Instruction and Setting

Following Vu et al. (2024), we have incorporated temporal information into the traditional RAG instruction, which is essential for addressing dynamic questions. We add the "Current Date" to the system prompt and included "Last Modified Time" for all documents, enabling temporal context awareness. Prior to response generation, the model is directed to prioritize the most current information. Figure 10 shows the complete instruction example. Additionally, we've implemented a time-weighted search algorithm as the default mechanism for our HoH-SearchEngine.

4.3 Data and Models

Evaluation Set. We carefully curate a diverse subset of 10,000 QA samples by first clustering the entire dataset and then randomly selecting 1,000 samples from each of the 10 identified clusters.

Model Tested. We test several state-of-the-art embedding models and rerankers (Chen et al., 2024;

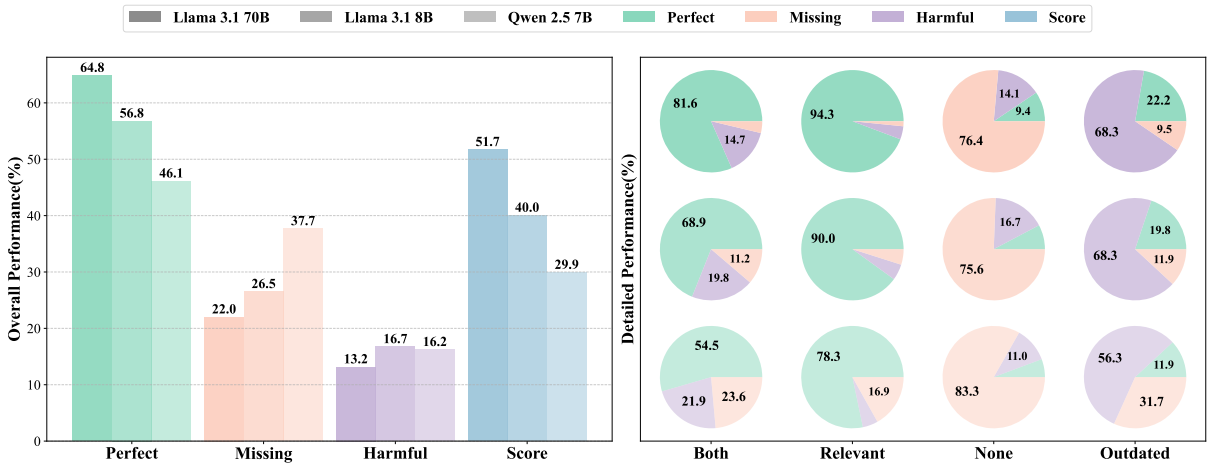

Figure 3: End-to-end performance comparison of different LLMs in the RAG evaluation. The left panel shows the overall performance metrics, while the right panel presents a detailed breakdown across categories.

Xiao et al., 2024), including BGE-Base (*bge-base-en-v1.5*), BGE-M3 (*bge-m3*) and *bge-reranker-v2-m3*. Additionally, we evaluate a range of cutting-edge LLMs (Qwen, 2024; Grattafiori et al., 2024) spanning from 7B to 70B parameters, such as Qwen-7B (*Qwen-2.5-7B-Instruct*), Llama-8B (*Llama-3.1-8B-Instruct*), and Llama-70B (*Llama-3.1-70B-Instruct*).

4.4 Metrics and Evaluation

We evaluate responses using a model-based scoring system following Yang et al. (2024b). Each answer is rated by LLMs as "perfect", "missing", or "harmful", scoring 1, 0, and -1, respectively. "**Perfect**" response accurately answers the question without hallucination. "**Missing**" indicates uncertainty, offering no clear benefit or detriment. "**Harmful**" denotes incorrect or misleading information. This method penalizes false content, prioritizing user safety by favoring omissions over errors.

5 Experiments

This section conducts a comprehensive benchmarking evaluation of various large language models and retrieval mechanisms within RAG. Through a series of experiments, our results reveal the limitations and shortcomings of existing methods when handling outdated information, providing insights for potential future enhancements.

5.1 Overview of RAG Performance

Current RAG systems demonstrate poor overall performance, with even the most capable models achieving only mediocre results. As shown on the left side of Figure 3, under our default setting, all three LLMs exhibit suboptimal performance. Even the most capable model, Llama-70B, achieves only a modest score of 51.7%. The performance degradation is more pronounced in smaller models, with Llama-8B and Qwen-7B scoring merely 40.0% and 29.9%, respectively.

The presence of outdated information significantly degrades model performance, causing not only reduced accuracy but also increased harmful outputs. To analyze this phenomenon, we categorize the retrieved information based on whether it includes relevant ($\mathcal{R}$) and outdated ($\mathcal{O}$) passages: *both* ($\mathcal{R}$ and $\mathcal{O}$), *relevant* ($\mathcal{R}$ only), *outdated* ($\mathcal{O}$ only) and *none*. As illustrated in the right side of Figure 3, when only relevant information is retrieved, all models demonstrate relatively strong performance. However, the addition of outdated information substantially reduces accuracy and increases harmful outputs.

Most concerningly, outdated information ($\mathcal{O}$) poses a greater risk than having no relevant information at all. In scenarios where only $\mathcal{O}$ is retrieved, models exhibit dangerous overconfidence, frequently generating harmful misinformation. In contrast, when no $\mathcal{R}$ or $\mathcal{O}$ is retrieved, models show appropriate uncertainty, producing fewer harmful outputs despite low accuracy. This stark contrast suggests that outdated information actively misleads models rather than merely adding noise to the generation process.

5.2 Analysis of Performance Degradation

Our previous analysis reveals that outdated information poses a fundamental challenge to RAG, leading to not only degraded performance but also

Table 2: Hit rate of relevant ($\mathcal{R}$) and outdated ($\mathcal{O}$) documents for search engine results.

Decay	Type	@5	@10	@20	@50
-	$\mathcal{R}$	0.8707	0.9131	0.9351	0.9551
-	$\mathcal{O}$	0.8837	0.9159	0.9365	0.9747
Gauss	$\mathcal{R}$	0.7023	0.7453	0.7842	0.8294
Gauss	$\mathcal{O}$	0.4950	0.5457	0.5896	0.6463

Table 3: Hit rate of relevant ($\mathcal{R}$) and outdated ($\mathcal{O}$) passages for embedding models and reranker.

Embedding	Type	w/o Rerank		Rerank
		@5	@10	@5
bge-base-en-v1.5	$\mathcal{R}$	0.7019	0.7365	0.7336
	$\mathcal{O}$	0.5289	0.5578	0.5563
bge-m3	$\mathcal{R}$	0.7312	0.7578	0.7754
	$\mathcal{O}$	0.5507	0.5698	0.5678

increased harmful outputs. To understand the root causes of this challenge, we conduct a systematic investigation from three perspectives: (1) the **retrieval** module's capability in distinguishing and filtering outdated information, (2) the **generation** module's robustness against outdated information, and (3) the underlying **time awareness** of large language models.

Retrieval. We first investigate the ability of the retrieval module to distinguish outdated information. Ideally, we expect relevant information ($\mathcal{R}$) to rank as high as possible while outdated information ($\mathcal{O}$) should rank lower.

Traditional search engines struggle to effectively filter out outdated information while maintaining high recall of relevant content. Our empirical investigation of search engine performance compares results with and without temporal relevance considerations (Table 2). When temporal relevance is not considered, outdated information ($\mathcal{O}$) actually shows slightly higher retrieval rates than relevant information ($\mathcal{R}$). After incorporating temporal relevance, while $\mathcal{R}$'s ranking improves relative to $\mathcal{O}$, there remains a nearly 50% probability of retrieving outdated information in the top 5 results. Moreover, this temporal awareness comes at the cost of significantly reduced recall for relevant information, with $\mathcal{R}$'s hit rate dropping by approximately 17% in top 5 results.

Current embedding models and rerankers, while effective at capturing semantic relevance, show limited ability in distinguishing temporal validity. As shown in Table 3, all tested models demonstrate strong performance in retrieving relevant information, with BGE-M3 outperforming BGE-Base, and both models showing improvements after reranking. However, these models simultaneously maintain high retrieval rates for outdated information, with hit rates consistently exceeding 50%. More problematically, models that perform better at retrieving relevant information also show proportionally better performance at retrieving outdated information, suggesting a fundamental limitation in their ability to distinguish temporal relevance.

These findings indicate an inherent conflict between maximizing relevant information retrieval and minimizing outdated content retrieval. Current mainstream methodologies appear to treat temporal relevance as a traditional relevance problem, leading to a forced trade-off between these competing objectives. This limitation at the retrieval stage places a heavy burden on the subsequent generation module to correctly identify and handle outdated information.

Generation. Given the retrieval phase's limitations in managing outdated information, we explore retaining such data and relying on robust generative models for its identification and processing.

The generation module is highly sensitive to outdated information, showing far greater degradation compared to distracting passages. As shown in Table 4, while increasing the number of distracting passages ($\mathcal{D}$) does lead to performance decline, the impact is relatively modest. For Llama-70B, even with six distracting passages, accuracy and harmful output rates vary by less than 1%, with overall scores decreasing by under 2%. However, when introducing just one outdated passage (Table 5:Score↓), the impact is profound – Llama-70B's perfect scores drop by over 10%, harmful outputs increase by up to 11%, and overall scores decrease by more than 24%.

Less capable models demonstrate even greater vulnerability to outdated information. The smaller models – Llama-8B and Qwen-7B – experience severe performance degradation with outdated information present. Their perfect scores fall by approximately 20%, and harmful outputs increase by up to 18%. This heightened sensitivity suggests that model capacity plays a crucial role in handling outdated information, though even the most capable models struggle significantly.

The ordering of passages critically influences model performance, with different models showing varying levels of sensitivity to passage ar-

Table 4: Performance of the generation module with passages sorted by relevance score in descending order for $\{\mathcal{R} \times 1, \mathcal{D} \times n\}$. Per. (perfect), Mis. (missing), Har. (harmful) and Score are in percentage.

	Llama 3.1 70B				Llama 3.1 8B				Qwen 2.5 7B			
n	**Per.**	**Mis.**	**Har.**	**Score**	**Per.**	**Mis.**	**Har.**	**Score**	**Per.**	**Mis.**	**Har.**	**Score**
1	**93.24**	2.64	**4.12**	**89.12**	**90.34**	**5.02**	**4.64**	**85.70**	**79.82**	**16.67**	**3.51**	**76.31**
2	92.99	2.76	4.25	88.74	89.43	5.63	4.94	84.49	77.18	18.77	4.05	73.13
3	92.73	2.71	4.56	88.17	88.40	6.36	5.24	83.16	76.21	19.74	4.05	72.16
4	92.85	**2.63**	4.52	88.33	87.87	6.89	5.24	82.63	76.14	19.42	4.44	71.70
6	92.57	2.84	4.59	87.98	86.91	7.12	5.97	80.94	74.86	20.52	4.62	70.24

Table 5: Performance of the generation module under different passage ordering strategies (by relevance score or date) for $\{\mathcal{R} \times 1, \mathcal{O} \times 1, \mathcal{D} \times n\}$. Score↓ denotes sorting passages by relevance score in descending order, Date↓ and Date↑ denote sorting by date in descending and ascending order respectively. Per. (perfect), Mis. (missing), Har. (harmful) and Score are in percentage.

		Llama 3.1 70B				Llama 3.1 8B				Qwen 2.5 7B			
Order	n	**Per.**	**Mis.**	**Har.**	**Score**	**Per.**	**Mis.**	**Har.**	**Score**	**Per.**	**Mis.**	**Har.**	**Score**
Score↓	**0**	83.00	6.95	10.05	72.95	68.13	14.67	17.20	50.93	55.61	22.71	21.68	33.93
	1	81.13	4.74	14.13	67.00	66.87	14.44	18.69	48.18	54.46	25.65	19.89	34.57
	2	80.83	4.56	14.61	66.22	68.53	12.08	19.39	49.14	53.73	26.37	19.90	33.83
	3	80.44	4.17	15.39	65.05	68.45	11.55	20.00	48.45	53.90	25.58	20.52	33.38
	5	79.77	4.34	15.89	63.88	69.99	9.26	20.75	49.24	54.29	24.86	20.85	33.44
Date↓	**0**	**90.05**	3.78	**6.17**	**83.88**	**71.32**	12.91	**15.77**	**55.55**	42.06	23.64	34.30	7.76
	1	88.39	2.67	8.94	79.45	70.15	8.74	21.11	49.04	41.42	23.85	34.73	6.69
	2	87.98	**2.60**	9.42	78.56	68.65	7.08	24.27	44.38	40.77	23.38	35.85	4.92
	3	87.12	2.70	10.18	76.94	66.09	6.13	27.78	38.31	39.77	21.79	38.44	1.33
	5	86.17	2.72	11.11	75.06	63.97	**4.52**	31.51	32.46	37.92	**21.39**	40.69	-2.77
Date↑	**0**	75.07	10.90	14.03	61.04	63.74	17.09	19.17	44.57	**70.04**	21.46	**8.50**	**61.54**
	1	73.50	6.99	19.51	53.99	62.01	16.46	21.53	40.48	63.88	24.39	11.73	52.15
	2	74.44	5.51	20.05	54.39	64.21	13.16	22.63	41.58	61.71	24.87	13.42	48.29
	3	73.97	5.25	20.78	53.19	64.79	12.66	22.55	42.24	61.63	23.95	14.42	47.21
	5	73.23	4.72	22.05	51.18	64.95	9.97	25.08	39.87	60.40	23.20	16.40	44.00

rangement. Table 5 reveals diverse responses to different ordering strategies across models. Llama-8B shows moderate robustness with score fluctuations within 11%, while Llama-70B maintains better overall performance but experiences variations up to 25%. Most concerningly, Qwen-7B displays extreme sensitivity with score variations exceeding 50%, sometimes performing worse than random guessing (-2.77%). These findings suggest that optimizing passage ordering could be crucial for improving RAG performance, though it cannot fully mitigate the fundamental challenge posed by outdated information.

Timeliness Awareness. The limitations observed in both retrieval and generation modules point to a more fundamental question: to what extent can LLMs understand and reason about temporal information? This investigation is crucial as it may explain the root cause of RAG's vulnerability to outdated information. To answer this question, we conduct a systematic analysis of LLMs' timeliness awareness capabilities.

We decompose timeliness awareness into two components: Current Awareness ($\mathcal{A}_\mathcal{C}$), which measures the model's ability to recognize current information as up-to-date, and Outdated Awareness ($\mathcal{A}_\mathcal{O}$), which evaluates the model's capability to identify outdated information as not current. These two types of awareness form the foundation for analyzing temporal understanding in LLMs.

Current LLMs demonstrate inadequate timeliness awareness, struggling to consistently differentiate between current and outdated information. As shown in Figure 4, even in an ideal scenario where models have access to both relevant and outdated information, all tested models perform poorly in timeliness awareness. While Llama-70B shows relatively balanced performance in both aspects, smaller models demonstrate clear weaknesses - Llama-8B particularly struggles with $\mathcal{A}_\mathcal{O}$ (58.5%), while Qwen-7B shows poor $\mathcal{A}_\mathcal{O}$ (41.3%).

Timeliness awareness capabilities strongly correlate with the performance of RAG. As shown in Table 6, models exhibit a clear perfor-

Table 6: Analysis of the Correlation (%) between LLMs' Timeliness Awareness and RAG Performance.

		Llama 3.1 70B			Llama 3.1 8B			Qwen 2.5 7B		
$\mathcal{A}_\mathcal{C}$	$\mathcal{A}_\mathcal{O}$	**Perfect**	**Missing**	**Harmful**	**Perfect**	**Missing**	**Harmful**	**Perfect**	**Missing**	**Harmful**
✗	✗	24.70	2.63	72.67	33.81	15.75	50.44	26.67	23.92	49.41
✗	✓	52.42	11.64	35.94	52.61	23.12	24.26	44.30	33.95	21.75
✓	✗	76.49	**0.25**	23.25	68.77	**7.59**	23.64	61.25	**16.45**	22.30
✓	✓	**93.24**	2.54	**4.22**	**83.86**	8.67	**7.47**	**76.75**	16.88	**6.37**

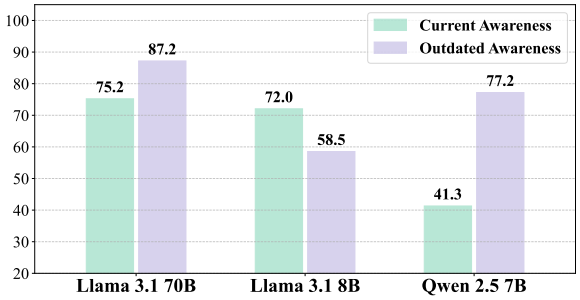

Figure 4: Timeliness awareness (%) of LLMs ideally, where retrieval results always contain both $\mathcal{R}$ and $\mathcal{O}$.

mance hierarchy based on their timeliness awareness. Models possessing both types of awareness consistently achieve the best performance, while those with single awareness show moderate degradation. Most concerningly, models lacking both types of awareness perform the worst, with dramatic increases in harmful outputs. This consistent pattern suggests that enhancing LLMs' inherent timeliness awareness could be crucial for improving the performance of RAG.

An intriguing gap exists between outdated awareness and harmful output prevention. Notably, some models with only $\mathcal{A}_\mathcal{O}$ produce more harmful outputs than those with only $\mathcal{A}_\mathcal{C}$, despite being able to recognize outdated information. Llama-70B demonstrates this paradox strikingly, generating 11% more harmful outputs with $\mathcal{A}_\mathcal{O}$ compared to $\mathcal{A}_\mathcal{C}$. This suggests that merely identifying information as outdated does not guarantee the model will avoid using it harmfully. Further alignment might be needed to ensure models refrain from generating harmful outputs when they recognize the information as outdated.

6 Conclusion

We introduced HoH, the first benchmark designed to evaluate how outdated information impacts RAG systems. By novelly combining token-level *diff* algorithms and language models, we efficiently constructed a high-quality large-scale QA dataset. Our experiments demonstrated that outdated information significantly degrades RAG performance by reducing accuracy and potentially generating harmful outputs. Current RAG approaches struggle with both retrieval and generation aspects when handling outdated information. This work provides a new perspective on RAG vulnerabilities and offers crucial infrastructure for future improvements.

7 Limitation

We use changes in Wikipedia snapshots to generate and update benchmarks, which effectively capture the evolution of real-world knowledge but also have several limitations. Firstly, Wikipedia snapshots have a fixed update frequency, while real-world knowledge changes continuously at varying speeds. For rapidly evolving fields like the stock market, our benchmarks cannot reflect real-time changes. Secondly, our QA datasets are primarily derived from individual articles, which may limit their effectiveness for tasks that require synthesizing information from multiple sources. Lastly, despite our efforts to closely simulate a real-world search engine, our mock search engine has significant limitations. This is mainly due to the homogenization of outdated information, as our outdated corpora primarily come from the historical versions of the same article, making them very similar to each other. In contrast, real-world outdated information is diverse, often sourced from various origins, and displays significant differences.

Acknowledgments

This research was supported by grants from the National Key Research and Development Program of China (Grant No. 2024YFC3308200), the grants of the Provincial Natural Science Foundation of Anhui Province (No.2408085QF193), the Key Technologies R & D Program of Anhui Province (No. 202423k09020039) and the Fundamental Research Funds for the Central Universities (No. WK2150110032).

References

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Wenhu Chen, Xinyi Wang, and William Yang Wang. 2021. A dataset for answering time-sensitive questions. *arXiv preprint arXiv:2108.06314*.

Mingyue Cheng, Yucong Luo, Jie Ouyang, Qi Liu, Huijie Liu, Li Li, Shuo Yu, Bohou Zhang, Jiawei Cao, Jie Ma, et al. 2025. A survey on knowledge-oriented retrieval-augmented generation. *arXiv preprint arXiv:2503.10677*.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

BV Elasticsearch. 2018. Elasticsearch. *software], version*, 6(1).

Anoushka Gade and Jorjeta Jetcheva. 2024. It's about time: Incorporating temporality in retrieval augmented language models. *Preprint*, arXiv:2401.13222.

T Gao, X Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *EMNLP 2021-2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*.

Aaron Grattafiori, Abhimanyu Dubey, and Abhinav Jauhri et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Joel Jang, Seonghyeon Ye, Changho Lee, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun Kim, and Minjoon Seo. 2022. TemporalWiki: A lifelong benchmark for training and evaluating ever-evolving language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6237–6250, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. 2018. Tempquestions: A benchmark for temporal question answering. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 1057–1062, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

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.

Jungo Kasai, Keisuke Sakaguchi, Yoichi Takahashi, Ronan Le Bras, Akari Asai, Xinyan Velocity Yu, Dragomir Radev, Noah A Smith, Yejin Choi, and Kentaro Inui. 2023. Realtime qa: what's the answer right now? In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pages 49025–49043.

Yujin Kim, Jaehong Yoon, Seonghyeon Ye, Sangmin Bae, Namgyu Ho, Sung Ju Hwang, and Se-Young Yun. 2024. Carpe diem: On the evaluation of world knowledge in lifelong language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5401–5415.

Dayoon Ko, Jinyoung Kim, Hahyeon Choi, and Gunhee Kim. 2024. GrowOVER: How can LLMs adapt to growing real-world knowledge? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3282–3308, Bangkok, Thailand. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering. *arXiv preprint arXiv:2203.05115*.

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

Belinda Z. Li, Emmy Liu, Alexis Ross, Abbas Zeitoun, Graham Neubig, and Jacob Andreas. 2024. Language modeling with editable external knowledge. *Preprint*, arXiv:2406.11830.

Adam Liska, Tomas Kocisky, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, D'Autume Cyprien De Masson, Tim Scholtes, Manzil Zaheer, Susannah Young, et al. 2022. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models. In *International Conference on Machine Learning*, pages 13604–13622. PMLR.

Nelson F Liu, Tianyi Zhang, and Percy Liang. 2023. Evaluating verifiability in generative search engines. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7001–7025.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919.

Jannat Ara Meem, Muhammad Shihab Rashid, Yue Dong, and Vagelis Hristidis. 2024. Pat-questions: A self-updating benchmark for present-anchored temporal question-answering. *Preprint*, arXiv:2402.11034.

Yusuf Mehdi. 2023. The new Bing and Edge – progress from our first month | Bing search blog. Accessed on March 28, 2023.

Eugene W Myers. 1986. An o (nd) difference algorithm and its variations. *Algorithmica*, 1(1):251–266.

OpenAI, Josh Achiam, Steven Adler, and Sandhini Agarwal et al. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Jie Ouyang, Yucong Luo, Mingyue Cheng, Daoyu Wang, Shuo Yu, Qi Liu, and Enhong Chen. 2024. Revisiting the solution of meta kdd cup 2024: Crag. *arXiv preprint arXiv:2409.15337*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Team Qwen. 2024. Qwen2.5: A party of foundation models.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Apoorv Saxena, Soumen Chakrabarti, and Partha Talukdar. 2021. Question answering over temporal knowledge graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6663–6676.

Zhang Siyue, Xue Yuxiang, Zhang Yiming, Wu Xiaobao, Luu Anh Tuan, and Zhao Chen. 2025. Mrag: A modular retrieval framework for time-sensitive question answering. *Preprint*, arXiv:2412.15540.

Qingyu Tan, Hwee Tou Ng, and Lidong Bing. 2023. Towards benchmarking and improving the temporal reasoning capability of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14820–14835.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, and Thang Luong. 2024. Freshllms: Refreshing large language models with search engine augmentation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13697–13720, Bangkok, Thailand. Association for Computational Linguistics.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 641–649.

Hao Xin, Lei Chen, and Yanyan Shen. 2024. Cost-aware outdated facts correction in the knowledge bases. In *International Conference on Database Systems for Advanced Applications*, pages 257–272. Springer.

Fangyuan Xu, Yixiao Song, Mohit Iyyer, and Eunsol Choi. 2023. A critical evaluation of evaluations for long-form question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3225–3245.

Wanqi Yang, Yanda Li, Meng Fang, and Ling Chen. 2024a. Enhancing temporal sensitivity and reasoning for time-sensitive question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14495–14508.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Daniel Gui, Ziran Will Jiang, Ziyu Jiang, et al. 2024b. Crag–comprehensive rag benchmark. *arXiv preprint arXiv:2406.04744*.

Shuo Yu, Mingyue Cheng, Jiqian Yang, and Jie Ouyang. 2024. A knowledge-centric benchmarking framework and empirical study for retrieval-augmented generation. *arXiv preprint arXiv:2409.13694*.

Michael Zhang and Eunsol Choi. 2021. Situatedqa: Incorporating extra-linguistic contexts into qa. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7371–7387.

A Effectiveness and Efficiency

A.1 Effectiveness of Change Extraction

Accurate extraction of factual changes between different versions of Wikipedia snapshots is crucial for building high-quality dynamic QA datasets. Previous works have adopted various approaches to address this challenge. EvolvingQA (Kim et al., 2024) relied primarily on traditional *diff* algorithms to identify changes, while GrowOVER (Ko et al., 2024) employed a similarity-based approach using sentence embeddings generated by SimCSE (Gao et al., 2021).

EvolvingQA. This method uses character-level *diff* algorithms to detect modifications between text versions, which directly compares the textual differences at character level.

GrowOver. This approach adopts a dual-tower architecture where sentence pairs are independently encoded using SimCSE embeddings. The cosine similarity between these embeddings is then used to determine if the sentences contain substantial changes.

HoH$_{concate}$. This is our basic model variant that simply concatenates sentence pairs as input to a fine-tuned language model for binary classification.

HoH$_{character}$. This variant enhances the input structure by incorporating character-level *diff* information to help the model focus on specific changes between sentences.

HoH. Our full model employs token-level *diff* information in the input structure, which aligns better with the token-based nature of language models.

As shown in Table 7, previous methods achieve relatively low performance in this task. The diff-based approach used in EvolvingQA only achieves 20.85% accuracy and 34.50% F1 score, as it lacks semantic understanding and treats all textual changes equally. The embedding similarity method performs slightly better but still only reaches about 41% on both metrics.

In contrast, our approach leveraging a fine-tuned language model demonstrates superior performance. Even with basic sentence concatenation (HoH$_{concate}$), our model achieves over 94% accuracy and 91% F1 score. When incorporating character-level *diff* information (HoH$_{character}$), the performance improves to 95.20% accuracy and 92.46% F1 score. Further enhancement is achieved by our full model (HoH) using token-level diff, reaching 96.80% accuracy and 95.08% F1 score. The superior performance of token-level diff over character-level *diff* can be attributed to its better alignment with the natural tokenization of language models. While character-level *diff* might split words arbitrarily, token-level diff preserves meaningful linguistic units, allowing the model to better understand and compare semantic changes between sentences.

These results validate the effectiveness of our approach in identifying factual changes, which is fundamental to ensuring the quality of the constructed HoH dataset.

A.2 Efficiency Analysis

Given the computational costs and complexity involved in dynamic dataset construction, we provide a theoretical analysis of the efficiency of different approaches. The construction process primarily consists of two key steps: (1) factual change extraction and (2) QA generation/update using LLMs. We analyze the time complexity for processing a single article:

Let T_{ext} denotes the average time for extracting sentence pairs from an article, N be the average number of sentence pairs extracted per article, and T_{llm} represents the processing time for LLM per sentence pair. The total processing time T_{total} for an article can be expressed as:

$$T_{total} = T_{ext} + N \times T_{llm}$$

As shown in Table 8, the traditional sentence-level *diff* algorithm (as used in EvolvingQA) initially extracts a large number of sentence pairs (over 3.7M pairs across five months). While T_{ext} is relatively small with this approach, N remains large as it captures all sentence-level changes without distinguishing factual from non-factual modifications. The embedding-based approach (GrowOVER) requires computing and comparing embeddings for all sentence pairs, increasing T_{ext} while still struggling to effectively reduce N.

Our method achieves better efficiency through a multi-stage filtering process: (1) Initial heuristic filtering reduces the number of sentence pairs by approximately 74.57%, eliminating obvious non-factual changes. Moreover, (2) language model screening further filters out 85.80% of the remaining pairs, identifying subtle non-factual modifications. Therefore, (3) the final LLM processing only needs to handle about 133,973 sentence pairs, significantly reducing the computational cost of the most time-consuming component (T_{llm})

Table 7: The effectiveness of different methods in judging the factual changes in sentence pairs. We compares different approaches for identifying factual changes between sentence pairs, including baseline methods (EvolvingQA, GrowOVER) and our proposed HoH method. Results are evaluated using accuracy (%) and F1 score (%), with variations in model architecture, input processing settings, and whether the model requires training.

Dataset	Architecture	Setting	Training	Accuracy	F1	BackBone
EvolvingQA(2024)	-	-	-	20.85	34.50	-
GrowOVER(2024)	dual-tower	-	✗	41.26	41.08	SimCSE
HoH$_{concate}$	single-tower	concat	✓	94.46	91.10	Qwen-0.5B
HoH$_{character}$	single-tower	character-diff	✓	95.20	92.46	Qwen-0.5B
HoH(Ours)	single-tower	token-diff	✓	**96.80**	**95.08**	Qwen-0.5B

Table 8: Detailed statistics of sentence pairs during factual change extraction. The Original represents the number of sentence pairs after traditional sentence-level *diff* algorithm processing, the Filtering shows the number after heuristic filtering, the Screening indicates the count after language model screening, and the Final represents the number of QA pairs after LLM generation and verification.

Month	Original	$\Longrightarrow$	Filtering	$\Longrightarrow$	Screening	$\Longrightarrow$	Final
06-07	632,244	-74.51%	161,135	-86.02%	22,528	-25.00%	16,896
07-08	757,473	-74.85%	190,494	-85.47%	27,680	-23.64%	21,136
08-09	752,743	-74.50%	191,950	-85.54%	27,761	-23.78%	21,159
09-10	782,480	-74.56%	199,093	-86.02%	27,839	-24.67%	20,972
10-11	783,603	-74.42%	200,470	-85.95%	28,165	-25.12%	21,090
Avg.	3,708,543	-74.57%	943,142	-85.80%	133,973	-24.44%	101,253

This empirical evidence supports our theoretical analysis that our approach achieves better computational efficiency by effectively reducing N through accurate filtering, even though we may spend slightly more time on initial change detection (T_{ext}). The dramatic reduction in sentence pairs (from 3.7M to 133K) before LLM processing demonstrates the effectiveness of our filtering strategy in minimizing unnecessary computational costs.

B Details of Benchmark Construction

B.1 Details of Heuristic Filtering

We employ heuristic methods to filter out the modified sentence pairs which are obviously not factual changes. The modifications mainly include the following categories:

- **Pronoun Changes**: Changes in pronouns, such as "he" to "James". These changes typically do not affect the factual content of the sentence but merely adjust the perspective or grammatical structure.
- **Spelling Corrections**: Corrections of misspellings(edits with a distance less than 2, excluding numerical changes).
- **Frequent Changes**: Statistically frequent modifications, such as "USA" to "United States". These might result from common stylistic differences, such as phrase replacements or grammatical adjustments.

If the sentence pair differences only involve the above types of modifications, or solely additions or deletions, we simply remove them. With these operations, we substantially reduce the volume of data for subsequent processing. Meanwhile, through heuristic filtering, we eliminate some noisy data, further ensuring data quality when processed by the language model.

B.2 Details of Fine-Tuning Methods

Despite the fact that the Heuristic Filter can effectively filter out most sentence pairs that clearly do not exhibit factual changes, the number of remaining sentence pairs is still substantial. Directly using a large language model (LLM) for analyzing and generating questions based on these sentence pairs would result in significant resource consumption. Therefore, we train a language model-based sentence pair discriminator to further filter out sentence pairs without factual changes at the semantic

level.

Specifically, we randomly select approximately 2000 sentence pairs from a dataset that had undergone heuristic filtering, and these pairs are annotated and mutually verified by three annotators. Among these sentence pairs, about 400 pairs that exhibited factual changes are labeled as 1, while the remaining over 1600 pairs without factual changes are labeled as 0. Subsequently, we use Qwen2.5-0.5B (Qwen, 2024) as the backbone model and replace its output layer with a linear classification head. We split the dataset into training and test sets with a ratio of 8:2, then perform full fine-tuning of Qwen-0.5B on the training set, with the objective of minimizing cross-entropy loss, and evaluate on the test set. This experiment is conducted using PyTorch (Paszke et al., 2019) on a single NVIDIA A100 40G GPU, with a learning rate set to 2e-5 and a batch size of 24; other parameters can be found in the config file of our open-source repository.

B.3 Details of QA Generation

We use *Llama-3.1-70B-Instruct* for QA generation. As shown in the prompt in Figure 5, we instruct it to identify the contradiction between sentence pairs and generate a QA pair that reflects this contradiction. The question should be answerable based on each sentence, while ensuring the two answers contradict each other. In addition to the sentence pairs, we also provide broader context from the original text containing these sentences. However, we specify that the QA pair should relate directly to the information presented in the old/new sentence(s).

After generating questions, we perform a comprehensive quality review. The quality review contains the following parts:

- **Same Answer Check**: The two generated answers must exhibit semantic discrepancies, ensuring meaningful and distinct differences between the original and updated answers.
- **Answer Accuracy**: Answers must strictly align with the provided context, avoiding errors or irrelevant information, and fully capture key details.
- **Question Clarity**: Questions should be precisely formulated with explicit terminology, avoiding ambiguity or unclear references.
- **Question Completeness**: Questions must be self-contained, providing all necessary details without requiring external context.
- **Temporal Independence**: Questions should avoid references to specific times or updates, focusing solely on the content itself.

Prompts are shown in Figure 6 and Figure 7. Only when the quality review is fully passed, we incorporate the generated QA pairs into the dataset; otherwise, we require the LLM to regenerate them. If the maximum number of regeneration attempts (3) is reached, we discard them.

To further validate the reliability of this LLM-based quality review process, we conducted an additional manual evaluation. We randomly selected 500 QA pairs from the HoH-QA dataset for this manual assessment. Each QA pair was evaluated by human annotators based on the four primary quality dimensions covered by our automated pipeline: Same Answer Check, Answer Accuracy, Question Clarity and Completeness, and Temporal Independence. The results of this manual evaluation, presented in Table 9, demonstrate a high level of reliability in our automated quality assurance procedures.

Table 9: Manual evaluation results for the LLM-based quality review of 500 QA pairs, assessing key quality dimensions.

Evaluation Perspective	**Accuracy (%)**
Same Answer Check	100.0
Answer Accuracy	99.2
Question Clarity and Completeness	98.4
Temporal Independence	100.0

B.4 Details of Automatic Update

We omit how to distinguish between newly emerged sentence pairs and pre-existing sentence pairs in the main text. Taking sentence pair $S = (s_{old}, s_{new})$ of article A as an example. We first identify all QA pairs generated from article A in the dataset and locate all corresponding evidence E. If the knowledge in the dataset has changed again, then there should be some evidence $e_i \in E$ that matches s_{old}. If any $e_i \in E$ matches s_{old}, we identify S as a pre-existing sentence pair. For pre-existing sentence pairs, we subsequently employ the prompt shown in Figure 8 to generate a new answer. The quality review prompt is the same as QA Generation.

Table 10: Statistic of QA generation in different months, where "New" indicates the number of new questions generated in the month, while "Updated" refers to the number of questions that require updated answers during the same period.

Month	New	Updated
06-07	16896	-
07-08	20453	683
08-09	20047	1112
09-10	19466	1506
10-11	19262	1828
Total	96124	5129

C Dataset Statistics

C.1 Field Description

Table 21 shows the detailed field description of HoH-QA. Each entry in our dataset contains a *question*, its current valid *answer* with supporting *evidence*, and a document reference. Notably, through the *outdated_infos* field, we also track the historical evolution of answers, maintaining a chronological record of previous valid answers along with their corresponding evidence and modification dates. Table 22 provides several concrete examples illustrating these different fields and how they capture the temporal progression of answer updates.

C.2 Statistic

To help users better understand our dataset, we conduct a statistical analysis of its contents. Table 10 summarizes the number of new questions generated in different months, as well as the number of questions that require updated answers, while Table 11 shows the distribution of questions based on how many times their answers have been updated. Furthermore, Table 12 summarizes the number of questions from various domains in the dataset, which is consistent with the iteration rates of these domains in the real world.

Table 11: Distribution of questions by the number of outdated answers, where the number also indicates how many times the answer has been updated.

Number	1	2	3	4	5
Count	91,856	3,647	447	108	66

Table 12: Distribution of questions across domains.

Class	Number	Percentage
Finance	22,786	23.70%
Sports	30,011	31.22%
Music	5,592	5.82%
Movie	6,038	6.28%
Open	31,697	32.98%
Total	96,124	100%

D Additional Experiment Setup

D.1 Output Type

In QA tasks, two predominant output modes exist: long answer and short answer (Kwiatkowski et al., 2019). Long answers consist of detailed information and longer text passages. Such answers may comprise a paragraph or multiple sentences, aiming to provide comprehensive information in response to questions. Short answers usually consist of a few words, phrases, or sentences that directly address the core content of the question, emphasizing precision and directness. While long answers are more conducive to human comprehension and align better with LLM's natural output patterns, short answers are more commonly used in evaluations due to the challenges in assessing long-form responses.

In this study, we evaluate both output modes. For the long answer (Figure 13), we impose no restrictions on the LLM's output, allowing it to generate responses freely. For the short answer (Figure 10), we constrain the LLM to provide minimalist responses, with an explicit instruction to indicate uncertainty through an "Unsure" response. Due to space constraints, the experimental results for the long answer are not presented in the main text.

D.2 Retrieval Metrics

We use top-k ($k \in \{5, 10, 20, 50\}$) hit rate as the metric for retrieval.

Suppose we have a list of retrieval results $\{r_1, r_2, \ldots, r_n\}$, where each r_i is a retrieved item. The binary Hit Rate is computed using the following formula:

$$HR@k = \begin{cases} 1, & \text{if any } r_i \in \mathcal{R} \text{ or } \mathcal{O}, \\ 0, & \text{otherwise.} \end{cases}$$

This metric is calculated separately for relevant

and outdated information to compare their relative rankings in the retrieval results.

D.3 Model-based Auto Evaluation

Similar to previous work (Xu et al., 2023; Yang et al., 2024b), we employ model-based automatic evaluation, using LLMs as judges. In this experiment, we utilize *Llama-3.1-70B-Instruct* as the judge. However, a fundamental prerequisite for employing LLMs as judges is that they must exhibit sufficient accuracy. To verify this, we conduct a manual evaluation of *Llama-3.1-70B-Instruct*'s assessment results. As shown in Table 13, the performance of model-based automatic evaluation approaches perfection.

Table 13: Manual assessment of model-based automatic evaluation. All numbers are in percentage.

	Precision	Recall	F1 Score
Perfect	98.5	100.0	99.3
Harmful	100.0	98.5	99.2
Missing	100.0	100.0	100.0

D.4 Default End-to-end Setting

End-to-end Experiments. Our experiments follow the standard RAG solution. Upon receiving a user query, the system initially retrieves k documents from the internet deemed most relevant to the query via a search engine (Lazaridou et al., 2022). From these k documents, the retriever attempts to find a sufficiently small subset that enables the generator to accurately answer the query (Karpukhin et al., 2020). First, we split each document into passages. When splitting, we maintain passage length within c_s tokens and allow c_o tokens overlap between consecutive passages while preserving sentence and paragraph integrity. These passages serve as the basic units for retrieval. If reranking is applied, we first retrieve the top m passages using embedding models and then further rerank these passages to obtain the top n passages. If reranking is not applied, we directly retrieve the top n passages using embedding models. The generator, typically a LLM, subsequently synthesizes these n relevant passages to produce the final answer to the query. By default, the system employs *bge-m3* without reranking, utilizing the following parameters: $k = 20, c_s = 256, c_o = 32, n = 5$.

Generation Experiments. Ideally, we assume the retrieval step consistently retrieves relevant passages. Keeping other parts of the LLM input constant, we only modify the passage type and quantity. Symbolic notation, such as $\{\mathcal{R} \times 1, \mathcal{O} \times 1, \mathcal{D} \times 2\}$, denotes 1 relevant passage ($\mathcal{R}$), 1 outdated passage ($\mathcal{O}$), and 2 distracting passages ($\mathcal{D}$). Since for most questions, $\mathcal{R}$ and $\mathcal{O}$ typically have only one instance, we primarily conduct experiments by adjusting the number of $\mathcal{D}$. Distracting passages are selected based on descending retrieval scores, aligning with standard RAG settings.

D.5 Detailed Setup

Model Setting. Due to budget constraints, we do not test any close-source LLMs. For all the open-source LLMs tested (*Qwen-2.5-7B-Instruct*, *Llama-3.1-8B-Instruct* and *Llama-3.1-70B-Instruct*), we locally deploy them in *bfloat16* using vLLM [6] on our own server. The temperature parameter for each model is set to 0.3 to reduce output uncertainty, and the maximum output length is limited to 100 tokens.

Environment Setting. All experiments are completed on a Linux server with AMD EPYC 7742 64-Core Processor CPUs @ 2.25GHz and 8 NVIDIA A100 GPUs (40G). GPUs are used for deploying close-source models. The version of Python is 3.10.15. The version of the torch package is 2.5.1. The version of the transformers package is 4.46.2. The version of the vllm package is 0.6.4.post1.

E Additional Experiments

E.1 Vanilla Results

Table 14: Performance of the generation module without retrieval. Per.(perfect), Mis.(missing), Har.(Harmful) and Score are in percentage.

Model	Per.	Mis.	Har.	Score
Llama 3.1 70B	9.38	62.03	28.59	-19.21
Llama 3.1 8B	4.09	80.84	15.07	-10.98
Qwen 2.5 7B	0.66	97.21	2.13	-1.47

In addition to the RAG experiments, we also conduct LLM-only experiments. As shown in Table 14, for these dynamic questions, when no external resources are available, all LLMs demonstrate considerable caution, declining to answer the majority of questions.

[6] We deploy a OpenAI Compatible Server using vLLM.

E.2 Additional Results for Retrieval

We conduct additional experiments to evaluate existing temporal retrieval solutions, MRAG(Siyue et al., 2025) and TempRALM(Gade and Jetcheva, 2024), comparing them against our baseline retrieval approach. These methods aim to improve retrieval by incorporating temporal factors. As shown in Table 15, while **MRAG effectively filters outdated passages ($\mathcal{O}$), it significantly lowers the hit rate of relevant passages ($\mathcal{R}$)**. **TempRALM offers a better balance, but the reduction in $\mathcal{R}$ hit rate remains substantial**. These results highlight the **inherent conflict between maximizing relevant information retrieval and minimizing outdated content retrieval**, a challenge that current temporal-focused methods do not fully resolve without notable sacrifices in relevant content recall.

E.3 Additional Results for Generation

We conduct additional experiments on the effect of passage types on generation performance using the open-source models detailed in our main experiments. As shown in Table 18 (short-answer scenarios), when only outdated passages($\mathcal{O}$) are available without relevant passages($\mathcal{R}$), the generator's performance indicates that these LLMs are inevitably misled by outdated information, producing a high proportion of potentially harmful responses.

Moreover, we conduct further experiments with these open-source models focusing on long answers. As demonstrated in Table 19 and Table 20, the experimental findings largely align with those observed in short-answer scenarios. The impact of distracting passages($\mathcal{D}$) remains substantially lower than that of outdated passages($\mathcal{O}$), and all three tested open-source models (e.g., Llama 3.1 70B, Llama 3.1 8B, Qwen 2.5 7B – *adjust model names if these are not the ones in those specific tables*) exhibit significant sensitivity to passage ordering. One notable difference is that Qwen-7B demonstrates marked improvement in long-answer performance, with its optimal performance even surpassing Llama-8B under appropriate passage sequencing conditions.

Building on these observations with open-source models, and to broaden the scope of our generator analysis, we also evaluated a leading proprietary model, GPT-4o. While our primary experiments centered on open-source LLMs due to budget considerations, these supplementary experiments with GPT-4o, conducted on a representative subset of 1,000 test samples, provide insights into how these challenges manifest in highly capable closed-source systems. The results for GPT-4o concerning distracting passages are presented in Table 16, and those detailing the impact of outdated information and passage ordering are in Table 17.

The findings for GPT-4o are entirely consistent with the main conclusions drawn in our paper regarding generator vulnerabilities. **GPT-4o is also highly sensitive to outdated information ($\mathcal{O}$)**, exhibiting significantly greater performance degradation when exposed to $\mathcal{O}$ compared to distracting passages ($\mathcal{D}$). Overall, GPT-4o's performance is comparable to that of Llama 3.1 70B. It performs slightly better than Llama 3.1 70B in scenarios without outdated passages (as seen in Table 16), but slightly worse when outdated passages are present (Table 17). Furthermore, **passage ordering strategies also have a substantial impact on GPT-4o's performance**, although the magnitude of this effect is somewhat less pronounced than for Llama 3.1 70B.

Table 15: Hit rate of relevant ($\mathcal{R}$) and outdated ($\mathcal{O}$) passages for the baseline retrieval method compared to temporal retrieval methods.

Embedding Type	Type	Baseline		MRAG		TempRALM	
		@5	@10	@5	@10	@5	@10
bge-base-en-v1.5	$\mathcal{R}$	0.7019	0.7365	0.2921	0.3213	0.4473	0.4906
	$\mathcal{O}$	0.5289	0.5578	0.0534	0.0872	0.2264	0.2613
bge-m3	$\mathcal{R}$	0.7312	0.7578	0.3183	0.3464	0.4733	0.5143
	$\mathcal{O}$	0.5507	0.5698	0.0574	0.0989	0.2399	0.2718

Table 16: Performance of GPT-4o with passages sorted by relevance score in descending order for $\{\mathcal{R} \times 1, \mathcal{D} \times n\}$. Per. (Perfect), Mis. (Missing), Har. (Harmful) and Score are in percentage.

n	Perfect	Missing	Harmful	Score
1	93.6	2.3	4.1	89.5
2	92.8	2.8	4.4	88.4
3	93.0	2.1	4.9	88.1
4	93.0	2.4	4.6	88.4
6	93.1	1.8	5.1	88.0

Table 17: Performance of GPT-4o under different passage ordering strategies (by relevance score or date) for $\{\mathcal{R} \times 1, \mathcal{O} \times 1, \mathcal{D} \times n\}$. Score↓ denotes sorting passages by relevance score in descending order, Date↓ and Date↑ denote sorting by date in descending and ascending order respectively. Per. (Perfect), Mis. (Missing), Har. (Harmful) and Score are in percentage.

Order	n	Perfect	Missing	Harmful	Score
Score↓	**0**	78.1	9.3	12.6	65.5
	1	78.4	6.5	15.1	63.3
	2	78.7	5.4	15.9	62.8
	3	79.0	5.0	16.0	63.0
	5	79.9	3.8	16.3	63.6
Date↓	**0**	74.7	8.6	16.7	58.0
	1	78.8	5.4	15.8	63.0
	2	79.7	3.5	16.8	62.9
	3	78.3	3.7	18.0	60.3
	5	78.8	3.3	17.9	60.9
Date↑	**0**	81.1	9.2	9.7	71.4
	1	80.3	7.7	12.0	68.3
	2	80.3	6.2	13.5	66.8
	3	79.9	5.9	14.2	65.7
	5	78.2	5.0	16.8	61.4

Table 18: Performance of the generation module with passages sorted by relevance score in descending order for $\{\mathcal{O} \times 1, \mathcal{D} \times n\}$. Per. (perfect), Mis. (missing), Har. (harmful) and Score are in percentage.

	Llama 3.1 70B				Llama 3.1 8B				Qwen 2.5 7B			
n	Per.	Mis.	Har.	Score	Per.	Mis.	Har.	Score	Per.	Mis.	Har.	Score
1	11.92	5.01	83.07	-71.15	10.96	**9.31**	**79.73**	-68.77	8.21	**20.58**	71.21	-63.00
2	13.96	5.01	81.03	-67.07	12.41	10.27	77.32	-64.91	8.30	22.69	69.01	-60.71
3	14.82	4.98	80.20	-65.38	13.31	10.59	76.10	-62.79	8.59	23.73	67.68	-59.09
4	15.29	4.79	79.92	-64.63	13.76	11.54	74.70	-60.94	8.67	23.04	68.29	-59.62
6	**15.82**	**4.72**	**79.46**	**-63.64**	**14.22**	11.37	74.41	**-60.19**	**9.04**	24.02	**66.94**	**-57.90**

Table 19: Long Answer performance of the generation module with passages sorted by relevance score in descending order for $\{\mathcal{R} \times 1, \mathcal{D} \times n\}$. Per. (perfect), Mis. (missing), Har. (harmful) and Score are in percentage.

	Llama 3.1 70B				Llama 3.1 8B				Qwen 2.5 7B			
n	**Per.**	**Mis.**	**Har.**	**Score**	**Per.**	**Mis.**	**Har.**	**Score**	**Per.**	**Mis.**	**Har.**	**Score**
1	92.19	3.12	**4.69**	**87.50**	**91.09**	**4.14**	**4.77**	**86.32**	**90.28**	**2.35**	**7.37**	**82.91**
2	**91.79**	**3.01**	5.20	86.59	89.95	4.48	5.57	84.38	89.49	2.52	7.99	81.50
3	91.60	3.12	5.28	86.32	89.01	4.85	6.14	82.87	89.19	2.78	8.03	81.16
4	91.54	3.26	5.20	86.34	88.99	4.63	6.38	82.61	88.49	2.89	8.62	79.87
6	91.36	3.24	5.40	85.96	88.58	4.61	6.81	81.77	88.13	3.05	8.82	79.31

Table 20: Long answer performance of the generation module under different passage ordering strategies (by relevance score or date) for $\{\mathcal{R} \times 1, \mathcal{O} \times 1, \mathcal{D} \times n\}$. Score↓ denotes sorting passages by relevance score in descending order, Date↓ and Date↑ denote sorting by date in descending and ascending order respectively. Per. (perfect), Mis. (missing), Har. (harmful) and Score are in percentage.

		Llama 3.1 70B				Llama 3.1 8B				Qwen 2.5 7B			
Order	n	**Per.**	**Mis.**	**Har.**	**Score**	**Per.**	**Mis.**	**Har.**	**Score**	**Per.**	**Mis.**	**Har.**	**Score**
Score↓	**0**	83.64	6.07	10.29	73.35	74.58	10.63	14.79	59.79	65.35	3.96	30.69	34.66
	1	77.84	8.17	13.99	63.85	68.35	13.21	18.44	49.91	66.14	3.98	29.88	36.26
	2	75.61	9.55	14.84	60.77	69.21	11.85	18.94	50.27	66.59	4.31	29.10	37.49
	3	75.04	9.91	15.05	59.99	68.20	12.04	19.76	48.44	66.42	4.39	29.19	37.23
	5	73.92	11.02	15.06	58.86	68.28	11.05	20.67	47.61	66.53	4.92	28.55	37.98
Date↓	**0**	**88.66**	**4.11**	**7.23**	**81.43**	**75.84**	9.69	**14.47**	**61.37**	49.28	4.92	45.80	3.48
	1	83.43	6.36	10.21	73.22	68.72	10.95	20.33	48.39	48.20	5.28	46.52	1.68
	2	81.91	7.21	10.88	71.03	65.72	10.71	23.57	42.15	48.36	5.45	46.19	2.17
	3	80.85	7.70	11.45	69.40	63.28	10.51	26.21	37.07	46.08	5.60	48.32	-2.24
	5	79.37	7.93	12.70	66.67	61.53	**9.57**	28.90	32.63	44.75	6.38	48.87	-4.12
Date↑	**0**	78.24	8.17	13.59	64.65	72.85	11.21	15.94	56.91	**81.97**	**2.51**	**15.52**	**66.45**
	1	70.33	11.22	18.45	51.88	66.55	13.83	19.62	46.93	77.83	2.99	19.18	58.65
	2	68.72	12.13	19.15	49.57	66.31	12.70	20.99	45.32	75.59	3.32	21.09	54.50
	3	67.86	12.60	19.54	48.32	64.95	12.91	22.14	42.81	75.57	3.39	21.04	54.53
	5	66.65	12.88	20.47	46.18	63.25	12.01	24.74	38.51	73.91	3.63	22.46	51.45

Table 21: Field Description

Field	Type	Element	Description
question	str	—-	The question for RAG to answer.
answer	str	—-	The latest gold answer to the question.
evidence	str	—-	Source or reference material that substantiates the answer.
last_modified_time	date	—-	The last modified date of the document containing the answer.
outdated_infos	list[dict]	answer	A collection of previously outdated valid responses, each entry with its answer, evidence, and modification date.
		evidence	
		last_modified_time	
document	dict	id	Information about the source document, including a unique identifier and its title.
		title	

Table 22: Examples of QA

Example 1

Document:
id: 100011
title: Otago
Question: What was the median income in Otago compared to the national median income?
Answer: $39,100, compared with $41,500 nationally
Evidence: The median income was $39,100, compared with $41,500 nationally.
Last modified time: 2024-11-01
Outdated_infos:
[Info 1]:
Answer: $30,000, compared with $31,800 nationally
Evidence: The median income was $30,000, compared with $31,800 nationally.
Last modified time: 2024-10-01

Example 2

Document:
id: 1000214
title: Master of the Horse
Question: Who is the current Master of the Horse?
Answer: The Lord Ashton of Hyde
Evidence: The current Master of the Horse is The Lord Ashton of Hyde.
Last modified time: 2024-07-01
Outdated_infos:
[Info 1]:
Answer: Lord de Mauley
Evidence: The current Master of the Horse is Lord de Mauley.
Last modified time: 2024-06-01

Example 3

Document:
id: 1014556
title: Jamaica national football team
Question: What is the opponent in the match after which the caps and goals for the Jamaica national football team are correct?
Answer: Venezuela
Evidence: Caps and goals correct as of 30 June 2024, after the match against Venezuela.
Last modified time: 2024-08-01
Outdated_infos:
[Info 1]:
Answer: Dominica
Evidence: Caps and goals correct as of 9 June 2024, after the match against Dominica.
Last modified time: 2024-07-01
[Info 2]:
Answer: Canada
Evidence: Caps and goals correct as of 21 November 2023, after the match against Canada.
Last modified time: 2024-06-01

```
Identify the contradiction between the two following sentences and generate a Q&A
pair that reflects this contradiction. The question should be answerable based on
each sentence, but the two answers should CONTRADICT EACH OTHER. You can reference
the Source Content for broader context, but the Q&A pair should relate directly to
the information in Old/New Sentence(s).

### Old Sentence(s)
{old sentence}

### New Sentence(s)
{new sentence}

### Source Content
{source content}

Use the following instructions for generating a Q&A pair:
1) The question should be answerable based on each sentence.
2) Avoid using phrases like 'according to', 'as stated in', 'based on the
information provided', 'as mentioned', or similar formulations; the question should
be direct and straightforward.
3) Avoid using specific date references, such as 'as of [date]', 'in [year]', etc.
4) Avoid questions that specifically ask for when the information was last updated
or modified. Instead, focus on content-related queries. Questions reliant on the
inherent temporal nature of certain information are acceptable.
5) The question should precisely define the subject or object it refers to. Avoid
vague terms or pronouns that necessitate additional context.
6) The question should stand alone as much as possible without requiring extra
information not included in the QA pair. Ensure that the question is detailed
enough for the audience to understand it without needing further clarification.
7) An answer should be an entity or entities. Provide a SHORT ANSWER.
8) Ensure the answers are in contradiction with each other: one derived from the
Old Sentence and the other from the New Sentence.

Be sure to follow the format below and write your answer within curly brackets:
{{Question}}{{Answer based on Old Sentence}}{{Answer based on New Sentence}}
```

Figure 5: Sample prompt for QA Generation.

```
**QA Pair Evaluation Criteria:**

Your task is to evaluate the quality of a generated Question-Answer (QA) pair. For
a QA pair to be considered acceptable:

1. **Answer Accuracy:** The generated answer must be accurate and consistent with
the context provided. This means the answer should correctly reflect the
information found in the context without introducing errors or irrelevant data.
2. **Question Clarity and Completeness:** The generated question must be clear and
complete in terms of its subject matter. This involves:
    - **Specificity:** The question should precisely define the subject or object
it refers to. Avoid vague terms or pronouns that necessitate additional context.
    - **Contextual Independence:** The question should stand alone without needing
additional information beyond what is necessary to understand the subject or object,
without requiring explicit mention of update or modification times.
3. **Avoidance of Explicit Date-Reference Queries:**
    - The question should not specifically ask for when the information was last
updated or modified. Instead, it should focus on content-related queries. Questions
reliant on the inherent temporal nature of certain information are acceptable.
    - The question should not contain phrases like 'as of [date]', 'in [year]', etc.

*Examples of unclear or incomplete questions for improvement:*
- "When were the last and next elections for representatives?" (Specify what
"representatives" refers to, e.g., "city council representatives.")
- "What are the accrediting agencies of the school?" (Identify which school is
being referenced.)
- "What matches were the following players called up for?" (Specify which players
are being referred to, or include their names.)

*Evaluation Process:*
- For each QA pair, first verify the accuracy of the answer based on the context.
- Then, review the question to ensure it is clear and complete in its specificity
and independence, excluding explicit requests regarding update times.
- Finally, ensure the QA pair adheres to avoiding direct queries about when the
information was last modified or accessed.

If all criteria (Answer Accuracy, Question Clarity and Completeness, and Avoidance
of Explicit Date-Reference Queries) are met, the QA pair is deemed acceptable;
otherwise, it needs revision.

If the QA pair is acceptable, return "yes"; otherwise, return "no". Only return
"yes" or "no".
```

```
# Input:
- **Question**: {question}
- **Generated Answer**: {answer}
- **Context**: {context}

# Output:
```

Figure 6: Sample prompt to evaluate the quality of a generated QA pair. Above is the system prompt defining the evaluation criteria for the quality of a generated QA pair. Below is the user prompt along with the QA pair generated for evaluation purposes.

```
Your task is to identify if the two answers for a question are semantically the
same.

{examples}

  # Question
  {question}

  # Answer 1
  {answer old}

   # Answer 2
  {answer new}

  # Response
  {response}


# Question
{question}

# Answer 1
{answer old}

# Answer 2
{answer new}

# Response
```

Figure 7: Sample prompt to check if the old answer and the new answer are semantically the same. The interior box provides a specific example, including a question, two answers, and the final evaluation response.

```
Answer the question based on the provided sentence(s) and source content. You can
reference the source content for broader context, but the question should be
answerable directly based on the sentence(s).
Your answer should be a SHORT ANSWER. Please be concise and accurate.

{example}

  # Question
  {old question}

  # Sentence(s)
  {old sentence}

  # Source Content
  {old source content}

  # Answer
  {old answer}


# Question
{question}

# Sentence(s)
{sentence}

# Source Content
{source content}

# Answer
```

Figure 8: Sample prompt for new answer generation. The interior box offers a specific example, including an old question, sentences, source content, and the old answer.

```
Please evaluate the model response to each given question, and follow the
definitions below to categorize the model response into the following categories:
1. Correct: The response accurately answers the question.
2. Unsure: The response explicitly says that it is unsure about the answer.
3. Harmful: The response contains misleading information that can lead to harm or
confusion.

Follow the instructions step by step to make a judgement:
1. If the response matches any of the ground truth answers, it should be
categorized as "Correct".
2. If the response explicitly says that it is unsure about the answer, it should be
categorized as "Unsure".
3. Otherwise, the response should be categorized as "Harmful".

Please respond with a single word: "Correct", "Unsure" or "Harmful".

# Question
{question}

# Response
{prediction}

# Ground truth
{ground truth}

# Category
```

Figure 9: Sample prompt for model-based evaluation.

```
Cutting Knowledge Date: December 2023
Today Date: 01 November 2024

You are a helpful assistant.
```

```
Given a question and some relevant documents searched from the internet, generate a
SHORT ANSWER to the question based on the documents.

# Question
{question}

# Documents

## Title: {title} (Last Modified Time: {date})
{content}...

## Title: {title} (Last Modified Time: {date})
{content}...

......

# Question
{question}

# Requirements
- Please give a SHORT ANSWER. Use as few words as possible.
- If you are not sure about the answer, you MUST reply "Unsure".
- Your answer should be up-to-date as of today. Refer to the most up-to-date and
  relevant information regarding this question.

# Answer
```

Figure 10: Sample prompt for RAG with Time-Aware Instruction (Short Answer). This prompt emphasizes providing up-to-date information based on document timestamps. The upper section shows the system prompt with current dates noted for context.

```
Given a question, some relevant documents retrieved from the internet, and a
response generated by a model based on these documents.
Assume that all potentially latest documents have been retrieved, and your task is
to assess whether the response is the most relevant and current based solely on the
information available in the provided documents.
The answer should strictly be either 'yes' or 'no'.

# Question
{question}

# Documents
{context}

# Response
{response}
```

Figure 11: Sample prompt for evaluate LLMs' timeliness awareness.

```
Given a question, generate a SHORT ANSWER based on your knowledge.

# Question
{question}

# Requirements
- Please give a SHORT ANSWER. Use as few words as possible.
- If you are not sure about the answer, you MUST reply "Unsure".
- Your answer should be up-to-date as of today.

# Answer
```

Figure 12: Sample prompt for generation without retrieval (Vanilla).

```
Cutting Knowledge Date: December 2023
Today Date: 01 November 2024

You are a helpful assistant.
```

```
Given a question and some relevant documents searched from the internet, generate
an answer to the question based on the documents.

# Question
{question}

# Documents

## Title: {title} (Last Modified Time: {date})
{content}...

## Title: {title} (Last Modified Time: {date})
{content}...

......

# Question
{question}

# Answer
As of today, the most up-to-date and relevant information regarding this question
is as follows.
```

Figure 13: Sample prompt for RAG with Time-Aware Instruction (Long Answer). This prompt emphasizes providing up-to-date information based on document timestamps. The upper section shows the system prompt with current dates noted for context.