TimeRAG: Enhancing Complex Temporal Reasoning with Search Engine Augmentation

Zhao Wang
Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
lilin22wz@gmail.com

Ziliang Zhao
Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
zhaoziliang@ruc.edu.cn

Zhicheng Dou*
Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
dou@ruc.edu.cn

Abstract

While Large Language Models (LLMs) augmented with search engines have achieved remarkable progress in open-domain question answering, their ability to adapt to a rapidly evolving world remains limited. A critical challenge lies in the need for complex temporal reasoning to answer real-world questions. Current Retrieval-Augmented Generation (RAG) methods primarily focus on retrieving the latest information but often fail to perform sophisticated temporal reasoning. To address this gap, we propose **TimeRAG**, a novel RAG framework designed to dynamically handle complex temporal reasoning tasks. TimeRAG operates through the iterative collaboration of two modules: (1) a temporal-semantic Query Decomposition (QD) module, which breaks down the original question into atomic time-event sub-questions to guide multi-step retrieval, and (2) a time-aware Answer Generation (AG) module, which analyzes temporal contexts, generates intermediate answers with confidence scores, and synthesizes the final answer upon reasoning completion. The system is trained in three stages: (1) time-aware supervised fine-tuning of the AG module, (2) imitation learning for the QD module to enhance temporal decomposition ability, and (3) reinforcement learning for end-to-end joint optimization to enhance temporal coherence across the entire system. Evaluations on three challenging benchmarks show that TimeRAG significantly outperforms existing methods, particularly on questions involving fast-changing real-world events and those grounded in false premises that require detection and correction of outdated or incorrect assumptions.

CCS Concepts

• **Information systems** → **Language models**.

Keywords

Time-sensitive Question Answering, Retrieval-Augmented Generation, Large Language Model

*Zhicheng Dou is the corresponding author.

CIKM '25, Seoul, Republic of Korea.

ACM ISBN 979-8-4007-2040-6/2025/11
https://doi.org/10.1145/3746252.3761425

ACM Reference Format:
Zhao Wang, Ziliang Zhao, and Zhicheng Dou. 2025. TimeRAG: Enhancing Complex Temporal Reasoning with Search Engine Augmentation. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25), November 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3746252.3761425

1 Introduction

Large Language Models (LLMs) have witnessed remarkable advancements in recent years. With their powerful capabilities in semantic understanding and natural language generation [1, 9], they have achieved significant breakthroughs in various knowledge-intensive tasks, particularly in open-domain question answering [7, 14]. However, existing LLMs exhibit notable limitations when handling time-sensitive questions, including a tendency to provide outdated information and generate temporally inconsistent or hallucinated responses [2, 24]. These limitations significantly compromise the reliability of LLMs in dynamic, real-world scenarios where information evolves rapidly. To address these challenges, Retrieval-Augmented Generation (RAG) systems, which integrate search engines to retrieve up-to-date external knowledge [10, 19], have emerged as a promising solution. By dynamically incorporating retrieved web evidence as contextual grounding, RAG systems enhance both the factual accuracy and temporal relevance of generated answers, and improve the reliability of LLM outputs in knowledge-intensive tasks [29].

Despite the growing success of RAG-based systems like Perplexity.AI[1] and Moonshot.AI[2], these systems still face significant challenges in replacing traditional search engines for real-world queries, particularly time-sensitive questions. Taking the seemingly easy question "Who was the previous head coach of Bayern Munich?" as an example, current LLMs frequently fail to properly identify and process the implicit temporal constraints ("previous"), often resulting in factually unreliable answers [17]. The complexity of time-sensitive questions stems from their inherent requirement for multi-fact temporal reasoning. In fact, most such questions are multi-hop: their answers cannot be obtained from a single retrieved document but must instead be synthesized from temporally dispersed evidence across multiple sources.

Recent studies have made strides in addressing the limitations of time-sensitive question answering. For example, Siyue et al. [30] introduced MRAG, a modular framework that enhances temporal retrieval through explicit temporal constraint extraction and semantic-temporal reranking. Vu et al. [32] proposed FreshLLMs,

[1]Perplexity.AI: https://www.perplexity.ai/
[2]Moonshot.AI: https://www.moonshot.cn/

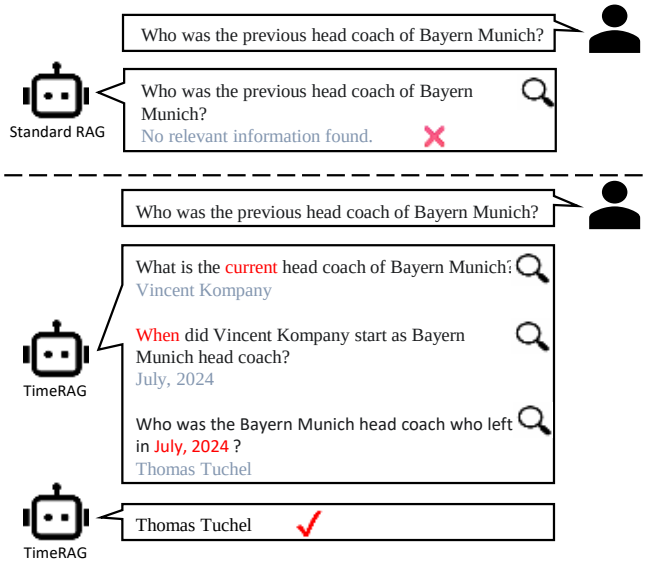

Figure 1: A comparative example between the Standard RAG and TimeRAG in the same question.

a few-shot prompting approach that improves temporal QA performance by integrating up-to-date search engine results. Yang et al. [36] developed temporal-aware embedding and granular contrastive reinforcement learning to boost LLMs' temporal sensitivity and reasoning abilities. Despite these improvements, current approaches primarily focus on optimizing information **freshness**, relying heavily on LLMs' inherent capabilities while neglecting structured temporal understanding. This fundamental limitation in existing RAG systems manifests in two critical ways: (1) inadequate comprehension of **implicit temporal constraints** (e.g., "previous," "since," "before") and **complex event relationships** (e.g., sequence, duration), and (2) difficulties in **integrating temporally dispersed information** for multi-step reasoning tasks.

To address these challenges, we propose **TimeRAG**, a novel RAG system designed to enhance complex temporal reasoning capabilities. TimeRAG answers questions through a dynamic collaboration between a temporal-semantic **Query Decomposition (QD)** module and a time-aware **Answer Generation (AG)** module. Specifically, the QD module orchestrates the reasoning process by iteratively generating atomic time-event subquestions, as shown in Figure 1. Concurrently, the AG module acts as a time-sensitive answer generator. For each generated subquestion, it processes the retrieved documents, generates the corresponding intermediate answer with confidence scores, and synthesizes the final response.

TimeRAG's overall training pipeline consists of three stages. First, in the **Supervised Fine-tuning** stage, the AG module learns to answer atomic time-event questions using document evidence. Second, in the **Imitation Learning** stage, we employ a powerful teacher model to interact with the AG module and generate high-quality temporal reasoning trajectories, which help the QD module learn effective query decomposition strategies. Finally, in the **Preference Alignment** stage, we leverage reinforcement learning for end-to-end joint optimization of the entire TimeRAG system. This process improves the AG module's ability to synthesize final answers, aligns the reasoning strategies between the two modules, and enhances temporal coherence across the entire system.

We evaluate TimeRAG on three representative open-domain temporal QA benchmarks, including FreshQA [32], RealtimeQA [18], and DailyQA [3]. The results demonstrate that TimeRAG outperforms existing methods, particularly on challenging fast-changing questions and false premise questions that require handling incorrect or outdated assumptions. Ablation studies confirm the robustness of TimeRAG across various retrieval configurations.

The main contributions of this paper include:

(1) We propose TimeRAG, a novel RAG system designed to enhance complex temporal reasoning capabilities.

(2) We design a temporal-semantic query decomposition module to generate atomic time-event subquestions and a time-aware answer generation module trained to interpret time information in retrieved documents and synthesize answers.

(3) We introduce a three-stage training pipeline combining supervised fine-tuning, imitation learning, and reinforcement learning for end-to-end optimization.

2 Related Work

2.1 Retrieval Augmented Generation

Retrieval-Augmented Generation (RAG) has emerged as an important paradigm to enhance the factual accuracy of LLMs by dynamically incorporating relevant information retrieved from external knowledge sources [19, 38]. This approach effectively mitigates critical limitations of standard LLMs, such as knowledge staleness and the generation of factual hallucinations [7, 39]. Early research on RAG primarily focused on end-to-end joint training approaches that tightly integrate the retriever and generator components, aiming to optimize the synergy between retrieval effectiveness and response generation quality [10, 19]. As the field has progressed, research has expanded to optimize key components of the RAG pipeline. Advances include query rewriting and expansion to better align user intent with retrievable content [23, 33], along with refined techniques for document compressing [35], denoising [5], and refinement [13, 16] to improve the relevance and efficiency of retrieved inputs. Moreover, instruction-tuned generators have enhanced RAG's ability to follow task-specific prompts and produce more controllable outputs [3, 4]. To address more complex questions, more recent studies have proposed iterative RAG systems [12, 15] that perform multiple retrieval steps, and sophisticated agent-based RAG systems [20, 21, 37] capable of autonomously planning and executing retrieval strategies. These advanced architectures have substantially improved the performance on tasks that require handling complex information needs. Despite these advancements, existing methods typically prioritize general information retrieval and generic query decomposition and lack sufficient capabilities for understanding the temporal information embedded within the documents, which makes them less effective in open-domain QA tasks that involve complex temporal reasoning.

[3] https://github.com/DailyQA/DailyQA

2.2 Time-Sensitive QA

Handling real-time knowledge in open-domain QA remains a persistent challenge for LLMs, as their static internal knowledge rapidly becomes outdated [18, 31]. To address this limitation, Time-sensitive Question Answering (TSQA) has emerged as a critical research area. TSQA requires models not only to interpret implicit or explicit temporal constraints within questions but also to generate accurate answers grounded in the temporally relevant information. Early TSQA efforts mainly focused on enhancing retrieval effectiveness for time-sensitive questions. For example, Kanhabua and Nørvåg [17] leveraged document timestamps and temporal expressions to rerank results, improving the relevance of retrieved content for temporally grounded questions. With the advent and widespread adoption of RAG, approaches that exploit dynamic knowledge sources, such as web search engines, to provide up-to-date contextual information for LLMs have gained significant attention [6]. Siyue et al. [30] introduced a modular architecture MRAG that explicitly identifies temporal constraints in questions and applies semantic-temporal reranking to retrieved passages, thereby enhancing the temporal alignment of supporting evidence. Similarly, FreshLLMs [32] adopt a lightweight, prompt-based strategy that questions web search engines at inference time to inject current information into LLM inputs, improving responsiveness to recent or evolving events. Beyond retrieval strategies, researchers have investigated model-level enhancements to boost temporal awareness and reasoning. Yang et al. [36] proposed a training framework combining Temporal Information-Aware Embeddings, which guide model attention toward temporal cues with Granular Contrastive Reinforcement Learning. This method significantly improves LLM performance across multiple TSQA benchmarks. Despite these advances, current approaches still face significant challenges, particularly in handling complex temporal reasoning and dynamically integrating information spanning diverse time periods.

3 TimeRAG: A RAG Framework for Complex Temporal Reasoning

TimeRAG is a novel RAG framework specifically designed to address the unique challenges of open-domain questions that require complex temporal reasoning and multi-hop retrieval of evolving factual information. In this section, we begin by outlining the overall architecture and inference pipeline in Section 3.1, highlighting the dynamic coordination among core components tailored to address complex temporal reasoning challenges. Next, we detail the framework's three-stage training approach: (1) time-aware supervised fine-tuning of the Answer Generation (AG) module to enhance the generation of time-sensitive answers (Section 3.2); (2) imitation learning to train the Query Decomposition (QD) module for effective temporal decomposition of complex questions (Section 3.3); and (3) reinforcement learning-based preference alignment to jointly optimize both QD and AG modules in an end-to-end manner, improving the overall consistency of the system (Section 3.4).

3.1 Overview

Figure 2 illustrates the overall architecture of TimeRAG, including a three-stage training pipeline and an inference workflow. It operates through the iterative interaction between two core components: the temporal-semantic Query Decomposition (QD) module and the time-aware Answer Generation (AG) module. In this section, we detail how they collaborate during the inference process.

3.1.1 The QD Module. This module is primarily responsible for interpreting the user's original complex question q and the evolving temporal reasoning state, which is dynamically updated based on the history of completed time-event subquestion–answer pairs from previous iterations. Leveraging this enriched context, the QD module determines the next logical step in the multi-hop reasoning process and generates a corresponding atomic time-event subquestion, explicitly formulated to pinpoint information valid at specific temporal points or intervals. For example, consider the question "Who was the previous head coach of Bayern Munich?" shown in Figure 1. Rather than attempting to answer it directly, the QD module first decomposes it into a simpler time-bounded subquestion, such as "Who is the current Bayern head coach?" Once the answer (e.g., Vincent Kompany) is obtained, it is incorporated into the updated temporal reasoning state. The QD module then uses both the original question and the new information to produce the next subquestion, e.g., "When did the current head coach take office?" This iterative process continues, with each subquestion guided by the progressively enriched temporal reasoning state, until sufficient information is gathered to accurately answer the original question.

3.1.2 The AG Module. The AG module serves as a time-aware generative model that processes information obtained through retrieval and answers the subquestions. Specifically, it receives atomic time-event subquestions from the QD module and analyzes the fine-grained temporal information embedded in documents retrieved from the search engine, including precise event timestamps, duration indicators, and historical contexts. By leveraging this temporal information, the AG module generates intermediate answers for each subquestion, along with confidence scores that reflect its certainty based on the available evidence, particularly regarding the temporal validity and relevance of the extracted facts. These answers and scores are then fed back to the QD module, providing updated facts and confidence signals to guide subsequent reasoning steps. This iterative process continues until the QD module determines that sufficient information has been gathered. At that point, the AG module enters a final synthesis stage, where it integrates the original user question with all accumulated subquestion–answer pairs to generate the final answer, ensuring temporal consistency and factual accuracy.

By implementing a well-structured inference pipeline comprising temporal-semantic decomposition, multi-step retrieval, time-aware generation, and chronological-coherent synthesis stages, TimeRAG is well-equipped to address complex temporal questions. This architecture enables the framework to effectively manage temporal constraints, identify and resolve fact changes over time, and perform multi-step reasoning across diverse sources and time periods. Consequently, TimeRAG can generate comprehensive answers to real-world questions that evolve over time. In the following sections, we will introduce how each module of TimeRAG is trained.

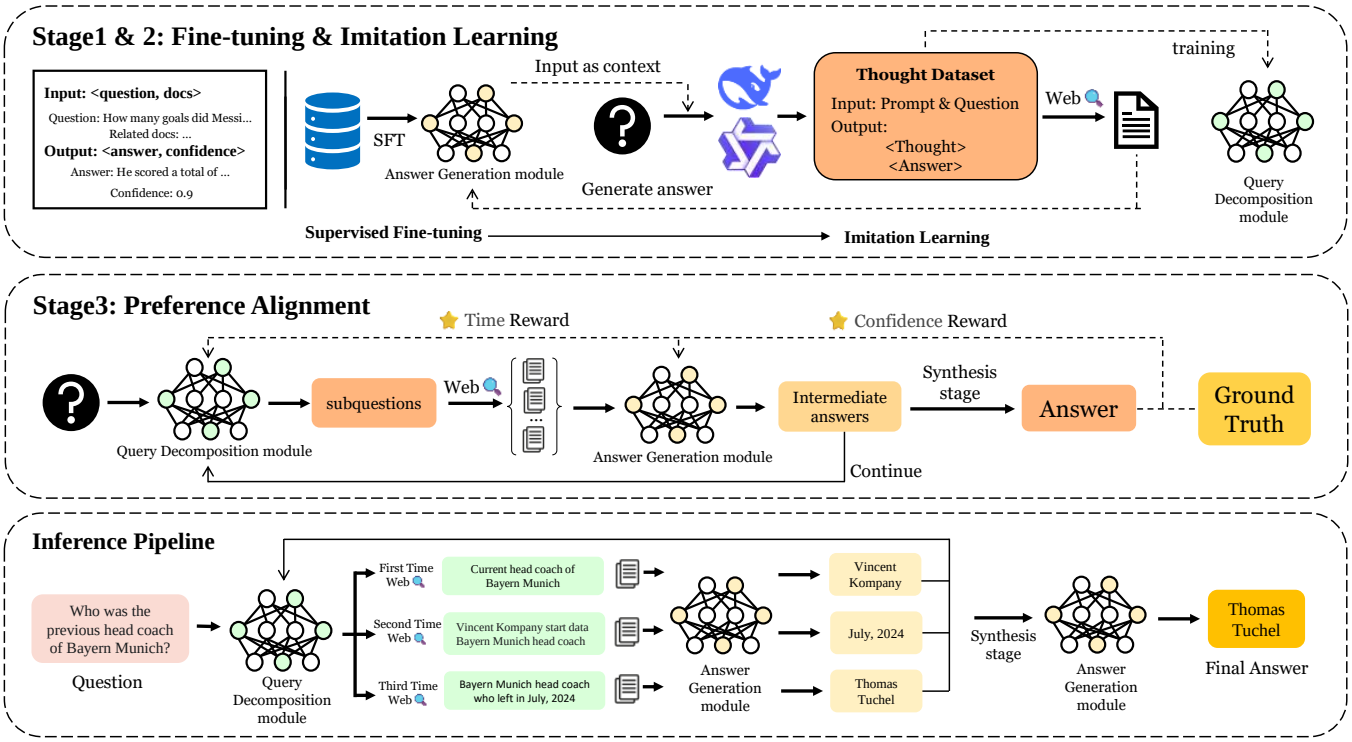

Figure 2: The Overall Framework of TimeRAG.

3.2 Fine-tuning Time-Aware Generation Model

At this stage of training, the primary goal is to enhance the AG module's temporal comprehension and its ability to effectively utilize time-sensitive information from retrieved documents. This capability is essential for enabling the AG module to support the subsequent iterative training of the QD module, which depends on the AG's capacity to handle intermediate reasoning steps. Supervised training in this phase targets two core competencies: (1) accurately answering atomic time-event subquestions by extracting and synthesizing relevant temporal information, and (2) predicting confidence scores that reflect the reliability of the generated answers. Reliable confidence estimates are critical for guiding the QD module's decision-making, allowing it to determine whether to trust or further verify an intermediate answer in the next reasoning step. Additionally, while the AG module will eventually be responsible for synthesizing the final answer, developing this capability is not the focus at this stage. Instead, final answer synthesis and end-to-end alignment of the TimeRAG pipeline are addressed in the preference alignment stage (Section 3.4).

Inspired by Dhingra et al. [2], we extract temporal facts from the Wikidata Knowledge Base (KB), retaining those associated with 10 time-sensitive relations identified in the TEMPLAMA dataset. Each fact is converted into a standardized quintuple format (s, r, o, ts, te), where s denotes the subject, r the relation, o the object, and ts and te represent the start and end times during which the fact (s, r, o) holds true. Based on these quintuples, we construct a comprehensive atomic time-event dataset for the supervised training of the AG module. This dataset comprises key components necessary for effective learning: atomic time-event questions, document contexts simulating retrieved evidence, ground-truth answers, and corresponding confidence scores.

To construct atomic time-event questions for supervised training, we design a procedure grounded in the temporal quintuples. For each quintuple (s, r, o, ts, te), we select a specific time point tr and generate questions q that explore the time–event relationship between the fact (s, r, o) and the reference time tr. These questions are formulated to either identify which event holds at a given time or determine when a particular event occurred. Based on the quintuple and the selected time tr, various question templates can be instantiated. For instance, given a fact represented as (subject: "Bayern Munich", relation: "head coach", object: "Thomas Tuchel", start time: "2023-03-24", end time: "2024-05-19"), and selecting a reference time tr as "March 2024", we can generate the question: "Who was the head coach of Bayern Munich in March 2024?"

For each atomic time-event question, we construct a corresponding document context to serve as the information source for the AG module. This context mainly consists of the natural language representations of the temporal quintuples. To better simulate the complexity of real-world search engine results, where relevant information is often mixed with irrelevant content, we deliberately introduce noise by injecting distractor events. These distractors are textual snippets derived from quintuples that share the same subject or relation but differ in object and time. For example, given the question "Who was Bayern Munich's head coach in the 2019–2020 season?", the distractor could be "Vincent Kompany took over as

Bayern Munich's head coach from Thomas Tuchel on July 1, 2025." Including such temporally misleading information helps the AG module learn to distinguish between relevant and irrelevant events.

For each question–context pair, we determine the correct target answer and calculate a confidence score c that reflects the answer's reliability given the context:

$$c = \frac{n_g}{n_g + w \cdot n_{\text{noise}}}, \tag{1}$$

where n_g denotes the number of golden facts, n_{noise} the number of noise facts, and w is a weighting factor that controls the influence of noise. To train the AG module to effectively handle unanswerable scenarios, such as when no relevant information can be retrieved, we also construct special cases where the context contains only noise facts and no golden facts. In these instances, the ground-truth answer is set to a predefined fallback: "There is insufficient information in the context to answer this question." with the corresponding confidence score fixed at $c = 0$.

We apply a pretrained Seq2Seq model as the backbone of the AG module and fine-tune it in a supervised manner on the constructed synthetic dataset. The model takes the concatenation of the question q and the context d as input, while the target output is a text sequence comprising the correct answer a and its corresponding confidence score c. The training objective is the standard cross-entropy loss between the model's output and the target sequence.

3.3 Query Decomposition Training with Imitation Learning

During the Imitation Learning stage, our primary goal is to train the temporal-semantic QD module to accurately interpret users' complex time-sensitive questions along with the dynamically evolving temporal reasoning state. This training aims to enable the QD module to effectively perform temporal-semantic decomposition, breaking down these questions into a sequence of logically coherent atomic time-event subquestions, thereby guiding TimeRAG's multi-step temporal reasoning process.

Considering the remarkable capabilities of recent advanced reasoning models such as DeepSeek-R1 [9], particularly in complex logical reasoning, task decomposition, and instruction following, we are motivated to adopt an imitation learning approach to train our temporal-semantic QD module. This approach leverages a powerful teacher model as an expert to generate high-quality sequences of temporal-semantic decomposition and chronological reasoning behaviors specifically tailored for complex time-sensitive questions. Our training then focuses on enabling the QD module to accurately imitate the reasoning process and effectively reproduce these expert demonstrations, thereby enhancing its specialized temporal decomposition capabilities.

To effectively construct the training dataset for the imitation learning stage, we expand upon the development set of the real-time open-domain QA benchmark, FreshQA [32], specifically selecting a diverse and challenging set of time-sensitive questions. For each original question q in the set, we guide the teacher model to perform step-by-step reasoning that simulates how TimeRAG handles such questions. During this process, the teacher model is instructed to analyze the user's original question along with the current temporal reasoning state, and then generate the temporally-grounded subquestions the QD module is expected to produce at each step. This process also thoroughly documents the entire temporal decomposition trajectory and its resulting outputs.

To ensure the generated action sequences effectively lead to correct and temporally accurate answers, we further validate and filter them by using the already trained time-aware AG module. By simulating TimeRAG's complete workflow, we include each turn's original question q, reasoning state, and teacher model outputs in the final training dataset only if the entire simulated process ultimately produces the correct answer.

Finally, we train the temporal-semantic QD module on the constructed dataset by supervised learning. The model input comprises the original question q and the current temporal reasoning state, while the target output is the validated teacher demonstration sequence of temporal-semantic decompositions. Training optimizes the model by minimizing the cross-entropy loss between the QD module's predicted output and the teacher model's demonstration.

3.4 Temporal Preference Alignment

Although supervised fine-tuning equips the time-aware AG module with its temporal comprehension and generation capabilities, and imitation learning endows the temporal-semantic QD module with its specialized decomposition abilities, training them separately does not inherently guarantee optimal performance during the final answer synthesis, especially for complex temporal questions. To address this limitation and ensure that the overall TimeRAG system produces answers that are not only accurate and aligned with human preferences but also chronologically consistent and temporally precise, we introduce reinforcement learning during the preference alignment phase. This stage aims to jointly optimize both the temporal-semantic QD and time-aware AG modules in an end-to-end manner. This joint optimization ensures temporally-aligned coordination between the modules, leading to more coherent, human-aligned, and robustly time-sensitive final answers.

We treat the entire TimeRAG as a policy to be optimized and train it by the CLIP version PPO algorithm [28]. This algorithm leverages CLIP to regulate the magnitude of model updates. The overall loss function consists of three components:

$$L_t^{\text{ALL}} = E_t[L_t^{\text{CLIP}} - L_t^{\text{VF}} + L_t^{\text{ENT}}], \tag{2}$$

L_t^{CLIP} is the policy optimization objective at step t, defined as:

$$L_t^{\text{CLIP}} = \min(r_t \hat{A}_t, \text{clip}(r_t, 1-\epsilon, 1+\epsilon)\hat{A}_t), \tag{3}$$

where r_t is the ratio of the conditional generation probabilities under the new and old policies:

$$r_t = \frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)}, \tag{4}$$

where ϵ is a hyperparameter controlling the update range, and A_t is the advantage estimate at step t, computed via Generalized Advantage Estimation (GAE) [27]:

$$\hat{A}_t = \sum_{l=0}^{K-1} (\gamma\lambda)^l (R_{t+l} + \gamma V(s_{t+l+1}) - V(s_{t+l})), \tag{5}$$

where γ and λ are hyperparameters, V is the value network estimating expected rewards, and R_t is the reward at step t. $L_t^{VF}(\theta)$ denotes

the value function loss, which fits the value network by minimizing the squared error between predicted and actual rewards:

$$L_t^{\mathrm{VF}}(\theta) = (V_\theta(s_t) - R_t)^2, \tag{6}$$

$L_t^{\mathrm{ENT}}(\theta)$ is a bonus to encourage exploration of new policies.

To compute the above loss, a well-defined reward function R_t is required. Given our goal of optimizing both the efficiency of temporal-semantic query decomposition and the quality of time-aware answer generation, we design a reward mechanism composed of two components: a reward R_1 for the retrieval process and a reward R_2 for the final answer.

The retrieval process reward R_1 aims to incentivize the temporal-semantic QD module to generate more efficient temporal reasoning trajectories by minimizing redundant search and decomposition steps. The reward function is defined as follows:

$$R_1 = \begin{cases} 0, & s_t \neq \text{<EOF>} \\ \beta - \lambda_1 \cdot step, & s_t = \text{<EOF>} \end{cases} \tag{7}$$

where <EOF> represents the end-of-sentence token, and $step$ indicates the number of reasoning iterations needed to complete the question. β is a base reward term, and λ_1 is a penalty coefficient applied to each additional reasoning or search step, with its sign determined by the correctness of the model's final answer. By maximizing R_1, the model is encouraged to reduce the number of search iterations while maintaining sufficient reasoning capability, thereby improving overall efficiency.

The answer generation reward R_2 directly evaluates the quality of the answer produced by the AG module, along with a confidence score. Its primary objective is to encourage the model to produce accurate answers while ensuring proper calibration of its confidence estimates, particularly by penalizing incorrect answers that are assigned high confidence. The reward function is defined as follows:

$$R_2 = \begin{cases} 0, & s_t \neq \text{<EOF>} \\ \lambda_2 \cdot confidence, & s_t = \text{<EOF>} \end{cases} \tag{8}$$

where $confidence$ denotes the confidence score assigned by the AG module to the generated answer, and λ is a weighting factor whose magnitude varies depending on the correctness of the answer. To effectively penalize incorrect answers with high confidence, we set the absolute value of the weight for incorrect answers to be greater than that for correct answers, i.e., $|\lambda_{2,\text{wrong}}| > |\lambda_{2,\text{correct}}|$. This ensures that the penalty for incorrect answers significantly outweighs the reward for correct ones at the same confidence level.

4 Experiments and Results

4.1 Dataset and Metrics

We evaluate TimeRAG on three key open-domain QA benchmarks to comprehensively assess its ability to handle time-sensitive questions: (1) FreshQA [32] (evaluated on its test set) focuses on evaluating a model's ability to handle time-sensitive and potentially false-premise questions. The questions are categorized into four types based on their reliance on temporal information and rate of change: false premise (FP), never-changing (NC), slow-changing (SC), and fast-changing (FC). (2) RealtimeQA [18] targets questions about recent events, emphasizing a model's capability to process extremely up-to-date, real-time information. (3) DailyQA is a multi-domain dynamic QA benchmark where questions are automatically updated based on Wikipedia revision logs. It is designed to evaluate how well models continuously track and adapt to rapidly evolving information across diverse domains.

For evaluation metrics, we adopted criteria suited to the characteristics of each benchmark. For the test set of FreshQA, we employed the Strict version of Fresheval [32] to measure answer accuracy, which allows flexibility in expression to better reflect real-world open-domain QA settings. For Realtime QA and DailyQA, we reported both Exact Match (EM) and F1 score to capture the token-level overlap between model predictions and reference answers. In addition, to more accurately assess responses to real-time or evolving information, we leveraged GPT-4o-mini [11] as an automatic evaluator.

4.2 Baselines

We include two categories of baseline methods in our experiments to assess the relative advantages of TimeRAG:

Vanilla LLM: These models answer questions solely based on their internal parametric knowledge, without access to any external sources. They represent as the baseline for evaluating LLM performance in the absence of retrieval augmentation. We include several widely adopted LLMs spanning various model sizes and capabilities: Qwen2.5-7B-Instruct [26], Llama-3.1-8B-Instruct [8], DeepSeek-R1-Distill-Llama-8B [9], as well as larger and more powerful models such as DeepSeek-R1-671B [9] and GPT-4o [11].

RAG-based: These methods augment LLMs with external knowledge retrieval, leveraging information obtained from search engines to assist in generation. Based on their retrieval workflows and optimization goals, we further categorize them into the following representative variants:

(1) **Standard RAG**: The variant represents the typical retrieval-augmented generation paradigm, where the LLM is combined with a search engine and performs a single-turn retrieval to obtain the top-K documents relevant to the original question, which are then provided as context for generation. We evaluate the performance of DeepSeek-R1-671B [9] and GPT-4 [1] under this standard RAG setup, and also include the commercial model Moonshot-v1, which has built-in web search capabilities.

(2) **RAG with Self-Ask**: These methods employ the LLM to decompose the original question into a sequence of sub-questions, enabling iterative retrieval to support more effective multi-hop reasoning [25]. We implement Self-Ask-based retrieval workflows using DeepSeek-R1-671B and GPT-4, serving as baselines for evaluating advanced reasoning capabilities.

(3) **RAG with FreshLLM**: Designed for time-sensitive QA specifically, these methods prompt the LLM to integrate up-to-date information retrieved from search engines with few-shot prompt engineering [32]. We evaluate FreshLLM variants based on DeepSeek-R1-671B [9] and GPT-4 [1], offering a direct comparison to TimeRAG in terms of temporal awareness and responsiveness.

4.3 Implementation Details

We adopt DeepSeek-R1-Distill-Llama-8B [9] as the backbone model for both the QD and AG modules due to its optimal balance between

Table 1: Comparison of different methods across FreshQA, RealtimeQA, and DailyQA benchmarks. The best results are in bold and the second are underlined.

Method	FreshQA					RealtimeQA			DailyQA			Avg.
	FP	NC	SC	FC	avg.	EM	F1	Acc.	EM	F1	Acc.	
Vanilla LLM (w/o Retrieval)												
Qwen2.5-7B-Instruct	5.6	35.8	17.7	7.0	16.4	10.0	12.3	13.3	2.9	6.1	4.1	11.3
Llama-3.1-8B-Instruct	4.0	48.0	21.8	5.5	19.6	3.3	6.4	16.7	1.2	3.7	2.0	12.8
DeepSeek-R1-Distill-Llama-8B	8.8	37.4	17.7	4.7	17.0	3.3	8.5	16.7	1.2	3.4	5.7	13.1
DeepSeek-R1-671B	13.6	78.0	51.6	14.8	39.2	10.0	12.4	13.3	6.0	10.9	9.8	20.8
GPT-4o	37.6	78.0	52.4	10.9	44.4	6.7	13.6	16.7	3.7	7.7	8.9	23.3
RAG Workflow												
Moonshot-v1	13.6	74.0	69.4	38.3	48.6	33.3	53.8	63.3	29.2	35.7	38.2	50.0
DeepSeek-R1-671B	20.0	87.0	70.2	50.0	56.6	30.0	43.3	60.0	32.1	41.8	45.9	54.2
w/ Self-Ask	64.8	90.2	66.1	36.7	64.2	30.0	37.8	46.7	21.6	28.2	31.8	47.6
w/ FreshLLM	75.2	87.0	71.0	50.8	70.8	40.0	48.5	60.0	34.6	38.9	42.0	57.6
GPT-4	19.2	91.1	72.6	54.7	59.2	43.3	50.5	63.3	31.3	41.9	47.2	56.6
w/ Self-Ask	43.2	85.4	64.5	38.3	57.6	33.3	41.5	50.0	26.5	31.6	36.2	47.9
w/ FreshLLM	71.0	**94.4**	77.6	55.2	75.6	43.3	55.4	63.3	**36.6**	43.7	48.4	62.4
TimeRAG (Our)	**76.0**	91.9	**78.2**	**69.5**	**78.8**	**46.7**	**59.2**	**70.0**	35.3	**44.1**	**50.4**	**66.4**

strong performance and computational efficiency. For retrieval, we extract the top 10 organic results from Google Search to serve as external context. During the imitation learning stage, DeepSeek-R1 [9] is employed as the teacher model to generate high-quality demonstration data. For optimization, we use the AdamW optimizer [22] with an initial learning rate of 5e-5 and a batch size of 4. In the preference alignment stage, a sampling-based strategy is adopted, setting top-k to 10 and top-p to 0.95, which promotes diversity while maintaining response relevance. All training and inference processes are implemented using the HuggingFace Transformers framework [34], which provides flexible and efficient tools for large-scale language model development.

4.4 Main Results

Table 1 presents the performance of TimeRAG compared with various baseline methods on three open-domain QA benchmarks: FreshQA, RealtimeQA, and DailyQA.

Overall, TimeRAG achieves the highest average accuracy of 66.4%, demonstrating its effectiveness in handling time-sensitive QA tasks. On the FreshQA, TimeRAG performs best on false premise and fast-changing questions. With an overall accuracy of 78.8%, it proves effective in resolving conflicts and integrating time-sensitive information. On RealtimeQA, TimeRAG ranks first in EM, F1, and accuracy, highlighting its strength in leveraging real-time external information. On DailyQA, it achieves the highest F1 and accuracy, and the second-best EM, demonstrating stable and reliable performance on general time-sensitive questions. These results highlight TimeRAG's strengths in temporal reasoning and RAG, and further validate the effectiveness of our time-aware query decomposition strategy and joint training framework.

Comparison with Vanilla LLMs. As shown in table, the Vanilla LLMs that only rely on internal knowledge perform poorly across all benchmarks. Their performance is especially weak on time-sensitive question types with most models achieving accuracy below 20%. Even large-scale models like DeepSeek-R1-671B and GPT-4o show decent results on never-changing questions and slow-changing questions, but still fall short on false premise questions and fast-changing questions. This notable performance gap emphasizes the necessity of integrating external retrieval mechanisms to enable accurate, up-to-date responses.

Comparison with Standard RAG. Standard RAG methods clearly outperform Vanilla LLMs by leveraging external retrieval, demonstrating greater robustness when handling static information. However, their effectiveness remains limited on highly dynamic categories such as false premise questions and fast-changing question in FreshQA. This indicates that their retrieval strategies are still insufficient to fully capture complex temporal shifts. Although Moonshot-v1 achieves decent accuracy on RealtimeQA and DailyQA, showing some capability in accessing up-to-date information, it falls short on EM and F1, suggesting a lack of targeted optimization for precise and comprehensive answer generation.

Comparison with Self-Ask. RAG methods enhanced with the Self-Ask multi-hop query decomposition mechanism show some improvement over standard RAG on never-changing and slow-changing questions that require step-by-step reasoning. However, they still struggle with tasks involving complex temporal logic, especially in the fast-changing categories of FreshQA. Notably, even when applied to powerful models like DeepSeek-R1-671B and GPT-4, the integration of Self-Ask fails to fully address the challenges posed by time-sensitive questions. This suggests that general multi-hop decomposition strategies remain insufficient for temporal reasoning, lacking the necessary adaptation.

Comparison with FreshLLM. FreshLLM is specifically designed for time-sensitive question answering and significantly outperforms general-purpose methods, particularly when powered

Table 2: Ablation Study Results on FreshQA.

Model	FP	NC	SC	FC	Avg.
TimeRAG	76.0	91.9	78.2	69.5	78.8
w/o RL	74.4	91.1	75.0	64.1	76.2
w/o IL	71.2	89.4	72.6	62.5	73.8
w/o SFT	72.0	90.2	70.2	60.9	73.2

by GPT-4. On FreshQA, FreshLLM (GPT-4) achieves an impressive 94.4% accuracy on the never-changing category and maintains competitive performance on the slow-changing questions and fast-changing questions. It ranks second in EM, F1, and accuracy on RealtimeQA and DailyQA. These results demonstrate the effectiveness of explicitly modeling and optimizing for temporal sensitivity in enhancing the timeliness of QA systems. However, FreshLLM still falls short of TimeRAG on several highly dynamic categories, suggesting that temporal awareness alone is insufficient. It requires combining it with multi-step retrieval and iterative optimization to fully addressing real-world temporal challenges.

4.5 Ablation Studies

To prove the effectiveness of each component in TimeRAG, we conducted a series of ablation studies on the FreshQA benchmark. We evaluated the following four model variants:

- **TimeRAG**: The complete version of our system, including a supervised fine-tuned AG module, a QD module trained via imitation learning, and a final joint optimization stage using reinforcement learning for preference alignment.
- **TimeRAG w/o RL**: This variant removes the RL-based joint optimization stage. Instead, the QD and AG modules are trained independently using imitation learning and supervised fine-tuning.
- **TimeRAG w/o IL**: In this variant, the IL-trained QD module is replaced by a simpler one-step retrieval strategy, where questions are used directly for retrieval without decomposition. The AG module remains time-aware supervised fine-tuned.
- **TimeRAG w/o SFT**: This variant removes the supervised fine-tuning stage for the AG module. Instead, the AG is initialized from a base model.

The results of the ablation study are presented in Table 2.

The full TimeRAG model achieves the highest overall accuracy on the FreshQA benchmark, validates the effectiveness of its three-stage training pipeline. Ablation results reveal that removing any single component leads to a noticeable drop in accuracy, underscoring the critical role each module plays.

Notably, removing the SFT stage results in the most significant performance degradation, with overall accuracy dropping to 73.2%, and particularly severe declines in the slow-changing and fast-changing categories. This underscores the crucial role of time-aware SFT in enabling the AG module to accurately interpret temporally sensitive information within retrieved documents.

Similarly, when the IL-trained QD module is replaced with a simpler one-step retrieval strategy, accuracy drops significantly to 73.8%. This version effectively reverts to standard RAG, which often retrieves outdated or irrelevant information due to its lack of temporal reasoning. These results highlight the need for temporally-aware, multi-step query decomposition when dealing with questions involving changing facts. The IL-trained QD module helps address this by modeling temporal reasoning and improving the relevance of retrieved evidence.

Table 3: Experiments with Top-k Documents on FreshQA.

Model	FP	NC	SC	FC	Total
w/k=5 results	73.6	88.6	76.6	67.2	76.4
w/k=10 results	76.0	91.9	78.2	69.5	78.8
w/k=15 results	76.8	92.6	79.0	70.3	79.6

In contrast, removing the RL-based joint optimization stage leads to a smaller but still noticeable drop in performance, particularly in the fast-changing category and overall accuracy. This indicates that although the AG and QD modules remain effective when trained separately, the RL stage is crucial for aligning their strategies. By enabling end-to-end optimization, it bridges the gap between SFT and IL, improving the system's coherence, coordination, and temporal reasoning under uncertainty.

In summary, the three training stages in the TimeRAG framework are all essential. Supervised fine-tuning contributes to temporal understanding, imitation learning enables effective query decomposition, and reinforcement learning ensures strategy alignment. Their combined effect enhances the system's ability to effectively address the complex challenges of real-world.

4.6 Impact of Retrieval Length (Top-K)

To assess the impact of retrieval length on the performance of TimeRAG, we evaluated the model on the FreshQA benchmark using three different values for the number of retrieved documents: Top 5, Top 10, and Top 15. The results are presented in Table 3.

The results demonstrate that expanding the number of retrieved documents steadily improves TimeRAG's performance. Average accuracy increases from 76.4% with Top-5 retrieval to 78.8% with Top-10, and further to 79.6% with Top-15. This positive trend holds across all question categories. For example, accuracy on the most challenging false premise questions rises from 73.6% to 76.0% with Top-10 and 76.8% with Top-15. Even for the relatively straightforward never-changing questions, which already exhibit high accuracy, modest improvements are observed as retrieval depth grows.

These findings suggest that more retrieval context provides TimeRAG with more comprehensive contextual information, which is especially beneficial for complex, time-sensitive questions. In such cases, critical evidence is often dispersed across multiple sources, and retrieving a larger set of documents increases the likelihood of capturing the necessary golden facts. TimeRAG's time-aware iterative query decomposition and generation components enable it to extract and integrate relevant information from extended contexts. This design allows the model to benefit from additional evidence without being adversely affected by distraction.

It is worth noting that the performance gains exhibit diminishing returns, with the improvement from Top-10 to Top-15 (0.8%) being smaller than from Top-5 to Top-10 (2.4%). While our results indicate TimeRAG can effectively utilize up to 15 documents, we did not extend k further. In practical applications, the choice of an optimal

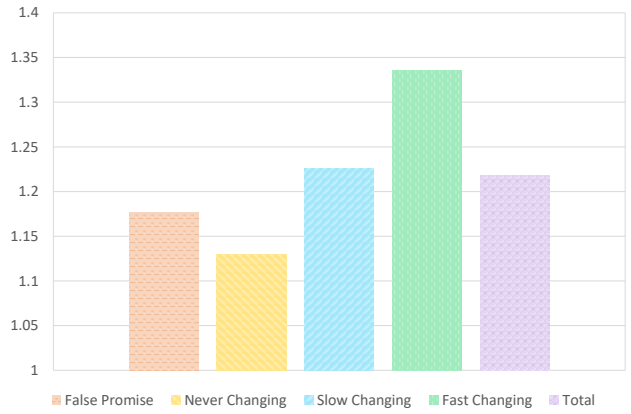

Figure 3: Average Retrieval Turns of FreshQA.

k involves a crucial trade-off between potential accuracy gains and the associated increases in retrieval latency and computational cost. Nonetheless, these findings underscore that providing a sufficient retrieval scope is an important factor in enhancing TimeRAG's ability to tackle complex real-world temporal questions.

4.7 Impact of Retrieval Turns

This section investigates the multi-hop reasoning capability of our proposed TimeRAG. We first analyze the adaptive behavior of the fully-trained TimeRAG model, presenting the average number of retrieval turns it naturally uses for different question types in Figure 3. Next, to assess the importance of this multi-step process, we evaluate the same model's performance by constraining its maximum number of retrieval turns at inference time to one and three, with results reported in Table 4. This approach allows us to directly measure the impact of reasoning depth without altering the model's training.

Figure 3 illustrates that TimeRAG dynamically adjusts the number of retrieval iterations based on the nature of each question in the FreshQA dataset. For relatively simple or temporally stable questions, such as those in the never-changing and slow-changing categories, the model typically completes reasoning with fewer retrieval turns. In contrast, for more complex and temporally sensitive questions, particularly in the false premise and fast-changing categories, TimeRAG performs substantially more retrieval iterations. This behavior suggests that the QD module can identify the complexity and temporal sensitivity of a question, initiating additional retrieval turns as needed.

The critical role of multi-hop reasoning is quantified in Table 4. While the full TimeRAG model achieves 78.8% accuracy, constraining it to a single retrieval turn causes a significant drop to 69.8%, particularly on complex types like false premise (69.6%) and fast-changing (57.0%) questions. This demonstrates the insufficiency of single-pass retrieval. Notably, increasing the limit to three turns largely restores performance to 77.2%. This recovery aligns with the average of approximately three turns required for complex questions as shown in Figure 3, indicating that a moderate number of retrieval iterations is sufficient to address most of the reasoning challenges in FreshQA.

Table 4: Experiments with Maximum Retrieval Turns.

Model	FP	NC	SC	FC	Total
TimeRAG	76.0	91.9	78.2	69.5	78.8
w/1 Turn	69.6	86.2	66.9	57.0	69.8
w/3 Turns	75.2	91.9	77.4	66.4	77.2

In summary, the results presented in Figure 3 and Table 4 indicate that TimeRAG's multi-hop reasoning capability, enabled by its QD module through iterative retrieval and information integration, is highly beneficial for achieving strong performance on complex time-sensitive questions, particularly those in the false premise and fast-changing categories. Constraining retrieval turns significantly reduces the model's ability to carry out the necessary reasoning steps, leading to noticeable drops in accuracy. These findings underscore the importance of TimeRAG's architecture, where decomposition, retrieval, and generation are integrated to tackle the challenges of temporal question answering with precision.

5 Conclusion

This paper tackles the limitations of current LLMs and RAG systems in complex time-sensitive question answering, particularly for tasks that demand multi-step reasoning over evolving information. While existing approaches enhance freshness through retrieval, they often lack structured temporal understanding and fine-grained reasoning capabilities. To address these challenges, we propose TimeRAG, a novel framework that enables iterative collaboration between a temporal-sematic Query Decomposition (QD) module and a time-aware Answer Generation (AG) module. By decomposing complex questions into time-event subquestions and integrating evidence across multiple retrieval turns, TimeRAG supports robust and adaptive temporal reasoning. We further introduce a three-stage training pipeline that combines supervised fine-tuning, imitation learning, and reinforcement learning for end-to-end optimization. Experiments on FreshQA, RealtimeQA, and DailyQA demonstrate that TimeRAG consistently outperforms strong baselines, especially on questions involving recent events and multi-hop reasoning. Ablation and retrieval analyses underscore the importance of its modular design and iterative mechanism. In summary, TimeRAG provides a principled approach to temporal reasoning in retrieval-augmented systems, paving the way for more reliable and time-aware QA models in dynamic real-world settings.

Acknowledgments

This work was supported by Beijing Municipal Science and Technology Project No. Z231100010323009, National Natural Science Foundation of China No. 62272467, Beijing Natural Science Foundation No. L233008. The work was also done at Beijing Key Laboratory of Research on Large Models and Intelligent Governance.

GenAI Usage Disclosure

In this paper, GenAI is primarily employed for partial data synthesis in the methodology, with the relevant details explicitly stated in the main text. Additionally, while GenAI is not used to draft the manuscript, GPT-4o is utilized for error checking after manual completion, including grammar and tense.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] 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 (2022), 257–273.

[3] Guanting Dong, Keming Lu, Chengpeng Li, Tingyu Xia, Bowen Yu, Chang Zhou, and Jingren Zhou. 2024. Self-play with execution feedback: Improving instruction-following capabilities of large language models. *arXiv preprint arXiv:2406.13542* (2024).

[4] Guanting Dong, Xiaoshuai Song, Yutao Zhu, Runqi Qiao, Zhicheng Dou, and Ji-Rong Wen. 2024. Toward general instruction-following alignment for retrieval-augmented generation. *arXiv preprint arXiv:2410.09584* (2024).

[5] Guanting Dong, Yutao Zhu, Chenghao Zhang, Zechen Wang, Ji-Rong Wen, and Zhicheng Dou. 2025. Understand what LLM needs: Dual preference alignment for retrieval-augmented generation. In *Proceedings of the ACM on Web Conference 2025*. 4206–4225.

[6] Anoushka Gade and Jorjeta Jetcheva. 2024. It's About Time: Incorporating Temporality in Retrieval Augmented Language Models. *arXiv preprint arXiv:2401.13222* (2024).

[7] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997* 2 (2023), 1.

[8] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783* (2024).

[9] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948* (2025).

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

[11] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276* (2024).

[12] Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403* (2024).

[13] Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *arXiv preprint arXiv:2310.06839* (2023).

[14] Zhengbao Jiang, Frank F 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 the 2023 Conference on Empirical Methods in Natural Language Processing*. 7969–7992.

[15] Zhengbao Jiang, Frank F 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 the 2023 Conference on Empirical Methods in Natural Language Processing*. 7969–7992.

[16] Jiajie Jin, Yutao Zhu, Yujia Zhou, and Zhicheng Dou. 2024. Bider: Bridging knowledge inconsistency for efficient retrieval-augmented llms via key supporting evidence. *arXiv preprint arXiv:2402.12174* (2024).

[17] Nattiya Kanhabua and Kjetil Nørvåg. 2012. Learning to rank search results for time-sensitive queries. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. 2463–2466.

[18] Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A Smith, Yejin Choi, Kentaro Inui, et al. 2023. Realtime qa: What's the answer right now? *Advances in neural information processing systems* 36 (2023), 49025–49043.

[19] 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 (2020), 9459–9474.

[20] Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366* (2025).

[21] Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yutao Zhu, Yongkang Wu, Ji-Rong Wen, and Zhicheng Dou. 2025. Webthinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776* (2025).

[22] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).

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

[24] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. *arXiv preprint arXiv:2005.00661* (2020).

[25] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350* (2022).

[26] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 Technical Report. arXiv:2412.15115 [cs.CL] https://arxiv.org/abs/2412.15115

[27] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* (2015).

[28] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[29] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems* 36 (2023), 8634–8652.

[30] Zhang Siyue, Xue Yuxiang, Zhang Yiming, Wu Xiaobao, Luu Anh Tuan, and Zhao Chen. 2024. MRAG: A Modular Retrieval Framework for Time-Sensitive Question Answering. *arXiv preprint arXiv:2412.15540* (2024).

[31] Qingyu Tan, Hwee Tou Ng, and Lidong Bing. 2023. Towards benchmarking and improving the temporal reasoning capability of large language models. *arXiv preprint arXiv:2306.08952* (2023).

[32] Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, et al. 2023. Freshllms: Refreshing large language models with search engine augmentation. *arXiv preprint arXiv:2310.03214* (2023).

[33] Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. *arXiv preprint arXiv:2303.07678* (2023).

[34] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*. 38–45.

[35] Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2023. Recomp: Improving retrieval-augmented lms with compression and selective augmentation. *arXiv preprint arXiv:2310.04408* (2023).

[36] Wanqi Yang, Yanda Li, Meng Fang, and Ling Chen. 2024. Enhancing temporal sensitivity and reasoning for time-sensitive question answering. *arXiv preprint arXiv:2409.16909* (2024).

[37] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

[38] Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Jie Jiang, and Bin Cui. 2024. Retrieval-augmented generation for ai-generated content: A survey. *arXiv preprint arXiv:2402.19473* (2024).

[39] Yujia Zhou, Yan Liu, Xiaoxi Li, Jiajie Jin, Hongjin Qian, Zheng Liu, Chaozhuo Li, Zhicheng Dou, Tsung-Yi Ho, and Philip S Yu. 2024. Trustworthiness in retrieval-augmented generation systems: A survey. *arXiv preprint arXiv:2409.10102* (2024).