LONG^{2}RAG: Evaluating Long-Context & Long-Form Retrieval-Augmented Generation with Key Point Recall

Zehan Qi[1*], **Rongwu Xu**[1*],
Zhijiang Guo[2†], **Cunxiang Wang**[3], **Hao Zhang**[4], **Wei Xu**[1†]
[1]Tsinghua University [2]University of Cambridge
[3]Westlake University, [4]Nanyang Technological University
{qzh23, xrw22}@mails.tsinghua.edu.cn
[*] Equal contribution, [†] Corresponding authors

Abstract

Retrieval-augmented generation (RAG) is a promising approach to address the limitations of fixed knowledge in large language models (LLMs). However, current benchmarks for evaluating RAG systems suffer from two key deficiencies: (1) they fail to adequately measure LLMs' capability in handling *long-context retrieval* due to a lack of datasets that reflect the characteristics of retrieved documents, and (2) they lack a comprehensive evaluation method for assessing LLMs' ability to generate *long-form responses* that effectively exploits retrieved information. To address these shortcomings, we introduce the LONG^{2}RAG benchmark and the Key Point Recall (*KPR*) metric. LONG^{2}RAG comprises 280 questions spanning 10 domains and across 8 question categories, each associated with 5 retrieved documents with an average length of 2,444 words. *KPR* evaluates the extent to which LLMs incorporate key points extracted from the retrieved documents into their generated responses, providing a more nuanced assessment of their ability to exploit retrieved information. Our dataset is available at: https://github.com/QZH-777/longrag.

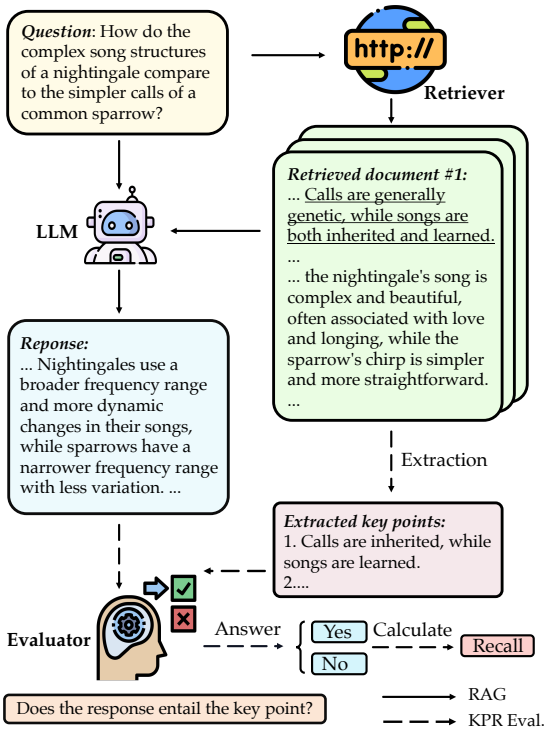


Figure 1: Illustration of the RAG and evaluation pipelines using *KPR*. We first extract the key points from the retrieved documents and compute the recall of these points in the response of the LLM with the help of an *Evaluator* (possibly another LLM), thereby enabling the evaluation of the response quality.

1 Introduction

Large language models (LLMs; Touvron et al. 2023; OpenAI 2024; Jiang et al. 2024) have demonstrated remarkable capabilities across a wide range of tasks. However, the fixed and finite nature of the knowledge embedded in LLMs presents limitations (He et al., 2022; Xu et al., 2024b). Retrieval-augmented generation (RAG), which incorporates external knowledge through search engines, represents a promising avenue for addressing this constraint (Borgeaud et al., 2022; Gao et al., 2023b).

Recent efforts in *long-context LLMs* (Xiong et al., 2023; Liu et al., 2024) leads to the packing of complete document content into LLMs to prevent information loss (Xu et al., 2023c; Gao et al., 2023b). This also challenges LLM's capability to handle long contexts. Several benchmarks are designed for evaluating long-context understanding (Xu et al., 2023c; Shaham et al., 2023; Liu et al., 2024). Nevertheless, none of them consider the characteristics of low signal-to-noise ratio and dispersed information distribution in retrieved documents of RAG, leading to *input-side deficiency*. Existing benchmarks fail to adequately measure LLMs' capability in handling ***long**-context RAG*.

Meanwhile, current assessment of LLMs within the realm of RAG mainly focuses on short answers (Chen et al., 2024; Zhang et al., 2024b),

Findings of the Association for Computational Linguistics: EACL 2024, pages 4852–4872

leaving a significant gap in evaluating LLMs' proficiency in generating ***long**-form responses with RAG*. This gap stems from the lack of a comprehensive evaluation method. For text generation, early automated metrics such as surface form matching (Lin, 2004; Banerjee and Lavie, 2005) and semantic representation comparison (Zhang et al., 2020; Yuan et al., 2021), face challenges with long-form content due to their inability to handle the diversity of potential outputs (Celikyilmaz et al., 2020; Krishna et al., 2021). Recent studies have explored the utility of LLMs for evaluation (Chiang and Lee, 2023; Liu et al.). However, these methods don't consider the retrieved documents within RAG. While some studies propose automated metrics for evaluating long-form generation in RAG (Es et al., 2023; Saad-Falcon et al., 2023), these metrics primarily focus on faithfulness, *i.e.*, whether the generated text is grounded in the retrieved documents. Therefore, no automated method exists for evaluating LLMs' exploitation of retrieved documents, representing a *output-side deficiency* of current RAG benchmarks.

To bridge this gap, we introduce LONG^{2}RAG, comprising 280 questions spanning 10 distinct domains and encompassing 8 question categories. Each question is associated with 5 retrieved documents, with an average length of 2,444 words per document. LONG^{2}RAG is collected with extensive care and offers several benefits. The questions posed in LONG^{2}RAG are both *intricate* and *practical*, requiring a comprehensive response. Furthermore, LONG^{2}RAG is carefully designed to *minimize the risk of data contamination*. To mirror the low signal-to-noise ratio and other characteristics prevalent in real-world scenarios, the associated documents originate from *authentic retrieval procedures*. To tackle the output-side deficiency, we propose *KPR*. As depicted in Figure 1, for each question, we automatically extract key points from the associated retrieved documents that directly contribute to answering the question. Subsequently, we evaluate the extent to which these key points are incorporated into the model's generated response, thereby assessing the effectiveness of LLMs in leveraging retrieved documents.

With LONG^{2}RAG and *KPR*, we extensively evaluate 9 state-of-the-art LLMs. We summarize our findings as follows:

- Closed-source LLMs represented by GPT-4o are more capable than open-source models, with the smaller open-source model (Phi-3-mini) being able to outperform the larger one of 72B (Qwen2).
- The model's capabilities show an overall decreasing trend as the input documents grow.
- The standard RAG procedure, *i.e.*, the truncation on retrieved documents, leads to a loss of information, resulting in weaker performance than RAG under long context.

We hope LONG^{2}RAG can facilitate the understanding of long-context RAG systems from multiple dimensions and facilitate the development of LLMs in exploiting retrieved information.

2 Related Works

2.1 RAG Benchmarks

Recent efforts in benchmarking RAG have primarily focused on two distinct evaluation objectives: retrieval and generation. Research on the ***retrieval*** aspect aims to assess the quality of the retrieved documents, considering factors such as retrieval relevance and timing (Lyu et al., 2024; Es et al., 2023; Saad-Falcon et al., 2023). Another line of work, where our study falls into, is concerned with the ***generation*** process (Chen et al., 2024; Zhang et al., 2024b; Stolfo, 2024), of which can also be categorized into short-form and long-form evaluation. In the former scenario, a succinct reference answer exists, and assessments predominantly rely on conventional metrics such as exact match (EM) and F1 score (Chen et al., 2024; Zhang et al., 2024b). However, this line of evaluation neglects the fact that people regularly use RAG for generation in real-world applications. For long-form evaluations, Gao et al. (2023a) evaluate the citation relevance during generation, while we evaluate the model's ability to identify key points, regardless of citations. Several recent studies evaluate the *precision*[1] of model-generated texts (Stolfo, 2024; Es et al., 2023; Saad-Falcon et al., 2023). Our research, on the contrary, adopts a ***recall*** perspective, assessing how well the generation captures key information present in retrieved documents. While a related work, namely CRUD (Lyu et al., 2024), also employs recall-like measurements, it focuses on utilizing one gold reference, which is not obtained by retrieval, to evaluate the generation. In contrast, our research is primarily oriented toward evaluating a

[1]Whether the claims within the generated text are grounded by retrieved documents.

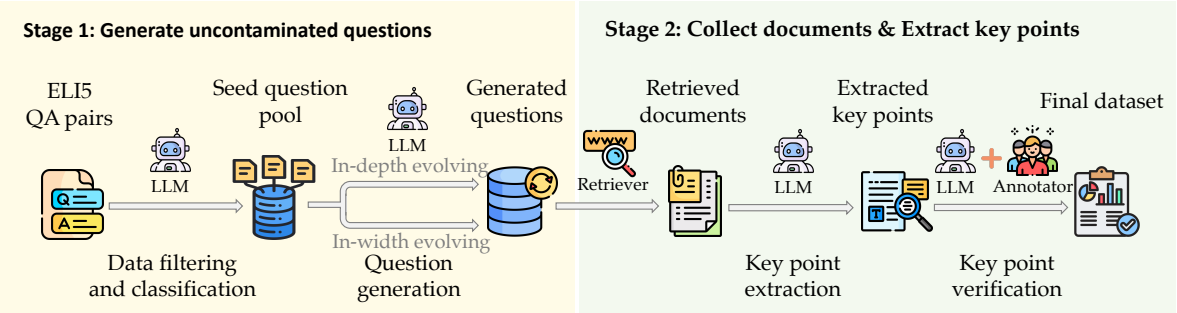


Figure 2: Overview of our dataset construction pipeline. The process comprises *two* main stages. In the first stage, we aim to generate uncontaminated questions by employing an LLM to filter questions from ELI5 and construct a seed question pool. By using two evolving techniques, new questions are generated. In the second stage, a search engine is utilized to procure documents for the RAG pipeline, where the key points are extracted automatically afterward. We finally employ a human-LLM collaborated verification task that result in our final dataset.

model's ability to utilize useful information from *multiple retrieved documents*.

2.2 Text Generation Evaluation

We classify long-form text generation evaluation into reference-based evaluation where gold answers are required and reference-free evaluation (Wei et al., 2024; Lyu et al., 2024). In the former case, methods employed to evaluate the similarity between the generation and the gold answer (Fan et al., 2019; Chiang and Lee, 2023) face challenges in acknowledging the legitimate range of potential appropriate answers (Krishna et al., 2021; Celikyilmaz et al., 2020). In addition, the evaluation outcomes may not align well with human judgments (Xu et al., 2023b). Motivated by these drawbacks, reference-free evaluation has attracted considerable interest, with some endeavors concentrating on assessing the coherence and relevance of the generation to specified *questions* (Fabbri et al., 2021; Krishna et al., 2022; Xu et al., 2023b, 2024a). Another line of literature explores the factuality of model generation by leveraging external *knowledge bases* (Stelmakh et al., 2022; Min et al., 2023; Wei et al., 2024). The work most closely related to ours is ProxyQA (Tan et al., 2024), which evaluates long-form generation through expert-designed *proxy questions*. We distinguish our approach by (1) focusing on the utilization of external knowledge in the RAG setup, (2) proposing an alternative *key point recall* measurement, and (3) largely reducing the need for human expert involvement in constructing key points.

3 Dataset Construction

In this section, we introduce the process of constructing LONG^{2}RAG. Overall, We leverage an automated pipeline[2] as illustrated in Figure 2. The generated dataset then went through human-LLM collaborated verification to ensure the quality.

To start with, we aim to create *questions* with the following properties:

- The questions are complex and *cannot* be easily answered by LLMs utilizing their parametric knowledge.
- The questions are practical and require a long-form answer.
- The questions are uncontaminated and thus less likely to be memorized by the LLMs.

Having prepared the questions, we utilize an automated approach to extract the *key points* from *retrieved documents* (obtained by leveraging a search engine), which will serve as the basis of our later evaluation. Finally, LONG^{2}RAG includes domain- and characteristic-diverse questions, paired with real-world retrieved documents and automatically extracted and human-verified key points *w.r.t.* to each question.

3.1 Question Generation

To ensure the practicality of the question, our starting point is ELI5 (Fan et al., 2019), a dataset collecting questions asked by users and corresponding answers from Reddit. However, questions in ELI5 face a potential risk of data contamination (Li, 2023; Golchin and Surdeanu, 2023), *i.e.*, being utilized as training material for LLMs. To address this drawback, we apply the Evol-Instruct (Xu et al., 2023a) method to further evolve existing questions in ELI5 to generate *fresh* ones.

To ensure consistency between questions acquired by Evolve-instruct and those sourced from

[2]We use GPT-4-Turbo as the $\text{LLM}_{\text{dataset}}$.

ELI5, we filter questions in ELI5 and create a seed question pool, which can be used to control the newly generated questions. Initially, questions within ELI5 are ranked based on the length of their corresponding answers, with the top 3,000 selected for further scrutiny. This subset then undergoes a filtering process utilizing an $\text{LLM}_{\texttt{dataset}}$, guided by specific criteria: (1) exclusion of common sense questions, (2) insurance of clarity and no ambiguity, (3) fulfillment of complexity requirements, and (4) solicitation for subjective opinions. Each question is assessed against the aforementioned criteria, with one point assigned for each criterion met. Questions accruing four or more points are retained, resulting in an *initial pool* comprising 1,445 questions. Subsequently, we categorize questions in the initial pool by $\text{LLM}_{\texttt{dataset}}$ into 8 categories as shown in Figure 3. For each category, a manual filtration process is employed to identify the top 12[3] questions align most closely with the defining attributes of this category, serving as the *seed question*. Finally, these 91 seed questions serve as our *seed question pool*.

Employing both the in-depth evolving method and the in-width evolving method in Evolve-Instruct, new questions are generated for each category. In addition to regulating the category of the generated questions, we also determine the domain in which these questions should be focused. A selection of 10 domains is designated to guide the question-generation process. Following the generation phase, a manual screening process is employed. Given the scale of the dataset under consideration, we decide to save 7 questions for each category-domain pair. This generation-screening process is repeated until a sufficient number of questions are obtained. So far, a dataset comprising 560[4] questions is created.

3.2 Document Collection

We adopt a question decomposition strategy leveraging the capabilities of $\text{LLM}_{\texttt{dataset}}$ to break down each question into several sub-questions to improve the quality of the retrieval process. We utilize two APIs, including the Google search engine and Serper search engine to retrieve pertinent documents for each subquestion. The top 5 documents obtained from each API are saved. Instead of using snippets, the complete content of each document is saved. For all retrieved documents from the subquery, a model[5] is used to rerank them, with only the top-5 most relevant to the query being retained. After this step, each question is accompanied by 5 most relevant real-world retrieved documents.

3.3 Key Point Extraction

We use an automated pipeline[6] to extract key points from retrieved documents using $\text{LLM}_{\texttt{dataset}}$. In LONG^{2}RAG, we define "key point" to be a concise, self-contained piece of information from the source documents that is both necessary and sufficient to formulate a complete and accurate answer to the given question, akin to the *score point in the grading process of an examinee's problem solution*. The detailed instructions for extracting the key points are listed in § B. To ensure comprehensive coverage, we conduct multiple rounds of extraction, each time prompting GPT-4 to consider previously overlooked information. All the extracted points for each question are then de-duplicated and aggregated by using the same LLM, leading to a more general expression of these key points. To ensure the aggregated points are complete and disjoint, the LLM also outputs the original points corresponding to each aggregated point. We check the aggregation results by seeing if an original point is in more than one aggregated point or not in any of them. Ultimately, this process yields a dataset comprising 28,611 key points.

We manually verify the effectiveness of this process on a subset of 20 randomly-sampled questions. For the documents associated with 10 randomly selected questions, we compare the model-extracted key points against human-identified key points. We calculate the recall of the model-extracted key points against the human-identified ones, achieving a 98.5% recall.

Human-LLM collaborated verification of the key points. We conduct a human-LLM collaborated verification to ensure the quality of the extracted key points. After sampling and observation, we find that most of the extracted points are not key points but only relevant to the question, with about 30% of the points meeting our definition. To reduce the cost of manual annotation, we apply the

[3]For subjective questions, it is insufficient to select 12 questions from the initial pool that meet the standards, we select 7 questions for this category instead.

[4]560 = 8 categories × 10 domains × 7 questions.

[5]Bge-Reranker-v2-Gemma is the model we use to rerank all retrieved documents based on their score with the query.

[6]Given the extensive length of retrieval documents with an average of 2,453 (as shown in Figure 12) and demand for expert knowledge, exclusive reliance on human effort for extraction is impractical.

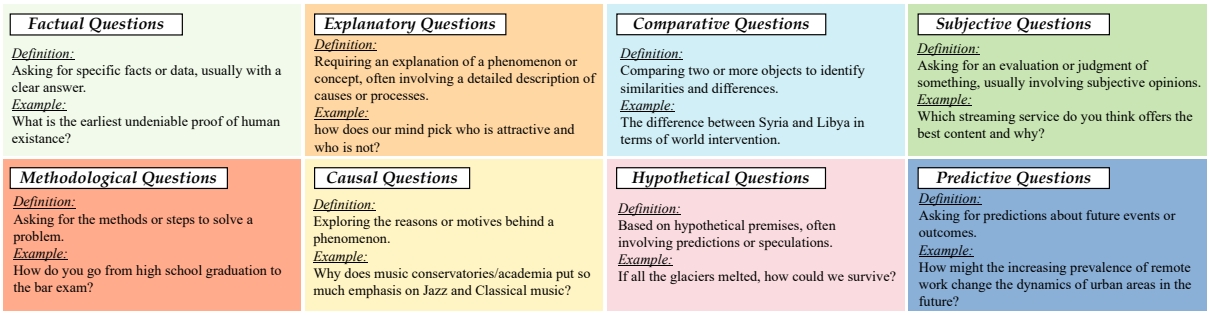


Figure 3: Detailed information about our defined question categories, including definitions and examples.

scoring function in AutoDS (Zhang et al., 2024a) to coarsely filter these points (see Appendix § A.2 for details). Using this score function, we calculate a score for each point to indicate its significance. We set 0.7 as the threshold, which demonstrates an 8.7% false positive rate by manually annotating a set of 20 questions with 1,051 key points. Details are deferred to Appendix § A.2. This results in a final 8,457 points remaining. Subsequently, we recruited 4 annotators to annotate the remaining points. They are encouraged to use the search engine to resolve any concepts that remain unclear during the annotation process. Their work is then subject to a spot-check to ensure that they adhere to the established guidelines.

Constrained by the annotation quality, we decided to use **2 out of the 4** annotators' annotation results (280 out of the 560 questions, the distribution of domains and question categories of the 280 results are shown in Appendix § A.1). The 2 annotators achieve an inter-annotator agreement (IAA) of 0.56, indicating moderate agreement. The IAA is assessed by the free-marginal multi-rater kappa (Randolph, 2005).

Final dataset. Initially, our automatic pipeline identified 3,651 key points linked to 280 questions. Following the human annotation process, we excluded points if they were not grounded by the document or did not meet the criteria for being considered a key point. Consequently, we retained 2,055 key points, which represents a 56.3% retention rate of the original points extracted by our automatic pipeline. While the dataset size of 280 questions might seem limited for an eight-category benchmark, it is comparable in scale to similar manually verified datasets like ProxyQA(Tan et al., 2024), which has 100 questions across 9 categories. We emphasize quality over quantity, selecting 280 high-quality questions from an initial 560 to ensure accuracy and reliability. Unlike other datasets that rely on scraped or extracted data, our manual annotation process minimizes contamination and inconsistencies, making it a robust resource for evaluating long-form text generation models.

4 *KPR*: a Newly Introduced Evaluation Metric

We propose the evaluation metric termed Key Point Recall (*KPR*) to evaluate to which extent the model exploits the retrieved documents. Consider a given question, denoted as q, for which d^q indicates the concatenation of retrieved documents and $\mathbf{x^q} = [x_1^q, x_2^q, \cdots, x_n^q]$ represents the set of all key points of the question. Let y denote the response generated by a model $\mathcal{M}$, where $y = \mathcal{M}(q \| d^q)$. We define an evaluation function $I(x_i^q, y)$ to assess the entailment relationship of key point x_i^q within the model's generation y:

$$I(x_i^q, y) = \begin{cases} 1 & \text{if } y \text{ entails } x_i^q, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

The evaluation function is implemented with an evaluator $\text{LLM}_{\texttt{evaluate}}$, using the prompt detailed in § A.2. Therefore, given a question dataset $\mathcal{Q}$, the *KPR* can be calculated as:

$$KPR(\cdot) = \frac{1}{|Q|} \sum_{q \in Q} \frac{\sum_{x \in \mathbf{x^q}} I(x, \mathcal{M}(q \| d^q))}{|\mathbf{x^q}|}, \quad (2)$$

where *KPR* calculates the average key point coverage in the question dataset.

5 Experiment

In this section, we apply LONG^{2}RAG and *KPR* to evaluate the performance of state-of-the-art LLMs.

5.1 Experimental Setup

Evaluated Models. We evaluate a vast array of both commercial API LLMs and open-source

LLMs. For API LLMs, we select 3 models, including GPT-4o, GPT-4-Turbo (OpenAI, 2024), and Claude-3-Sonnet. For open-source LLMs, we select 6 models of different sizes considering their parameters. We select Qwen2 models in small (< 7B), medium (~ 7B), and large size (~ 70B). We also incorporate popular LLMs including Phi-3-mini-128K (Abdin et al., 2024) and Mis(x)tral-Instruct (both 7B (Jiang et al., 2023) and 8*22B Sparse Mixture of Experts (SMoE) (Jiang et al., 2024) variants) in the evaluation.

For all models, we employ a straightforward approach to integrate the retrieved documents. We concatenated multiple retrieved documents directly, ensuring that clear separators were placed between each to maintain coherence. Subsequently, we appended a tailored prompt: *"Your answer should incorporate as many important points from the documents as possible that help in answering the question."*. This prompt was designed to encourage the generation of comprehensive, long-form responses. We configure all models using greedy decoding.

Selection of the Evaluator. In theory, any performance language model could serve as an evaluator $\text{LLM}_{\texttt{evaluate}}$ for its capability to assess textual entailment. However, when taking into account model performance, inference efficiency, and cost, we chose GPT-4o as our evaluator.

Research Questions. In order to conduct an extensive examination of the capabilities of LLMs and to evaluate the efficacy of our proposed LONG^{2}RAG benchmark, the following research questions have been formulated to guide our inquiry:

- **RQ1:** How well do prevalent LLMs exploit key information in long-context & long-form RAG?
- **RQ2:** What is the difference in capabilities for different domains?
- **RQ3:** What is the difference in capabilities for different question categories?
- **RQ4:** How does the length of documents in the context of RAG affect model performance?
- **RQ5:** If a document is truncated on the input side, what is the impact on the performance?
- **RQ6:** Does *KPR* favor longer generation?
- **RQ7:** Does the use of different models as evaluators maintain consistency in *KPR*?

Next, we will answer these questions one by one.

5.2 Main Results

The overall evaluation results of 9 LLMs are shown in Table 1. For **RQ1**, we conclude that:

- closed-source API LLMs generally outperform open-source ones, with GPT-4o being the most competent LLM;
- overall, in the case of QWen2, an increase in the size of the model results in a corresponding increase in performance;
- larger models do not always beat smaller ones, the smaller Phi-3-mini outperforms the 8*22B Mixtral and nearly approaches Qwen2 72B.

5.3 Results on Different Domains

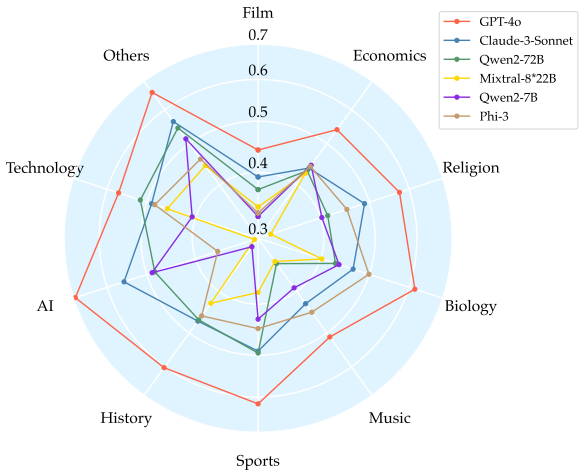


Figure 4: *KPR* of LLMs on different domains.

To answer **RQ2**, we present a radar plot that compares the model performance across domains in Figure 4. We discover that GPT-4o outperforms other LLMs (including closed-source ones) across all domains, by a large margin. Meanwhile, an interesting phenomenon is that all the models we evaluate do not cover the film domain very well. In addition, we observe that each model exhibits a degree of specialization in distinct domains. For instance, GPT-4o and Claude-3-Sonnet demonstrate superior performance on problems within the AI domain. In contrast, AI-related problems pose a challenge for Phi-3 and Mixtral, indicating a relative weakness in their domain-specific capabilities.

5.4 Results on Different Question Categories

Figure 5 depicts the detailed evaluation results within 8 question categories. In addressing **RQ3**, we have identified a pattern analogous to that observed in RQ2. However, the distinction here lies

Model	Size	Factual	Explanatory	Comparative	Subjective	Methodological	Causal	Hypothetical	Predictive	*Average*
API LLMs										
GPT-4o	N/A	**0.621**	**0.645**	**0.658**	**0.658**	**0.487**	**0.559**	**0.515**	**0.580**	**0.579**
GPT-4-Turbo	N/A	0.542	0.492	0.540	0.560	0.403	0.446	0.436	0.417	0.469
Claude-3-Sonnet	N/A	0.483	0.484	0.561	0.513	0.477	0.437	0.394	0.537	0.477
Open-source LLMs										
Qwen2-Instruct	72B	0.548	0.452	0.586	0.491	0.394	0.417	0.414	0.392	0.449
Mixtral-Instruct	8*22B[1]	0.482	0.509	0.425	0.425	0.303	0.336	0.315	0.385	0.383
Qwen2-Instruct	7B	0.379	0.462	0.470	0.464	0.422	0.478	0.361	0.360	0.416
Mistral-Instruct	7B	0.509	0.426	0.456	0.474	0.290	0.308	0.315	0.329	0.373
Qwen2-Instruct	1.5B	0.305	0.303	0.276	0.323	0.290	0.225	0.243	0.186	0.262
Phi-3-mini-128K	3.8B	0.488	0.465	0.483	0.514	0.409	0.393	0.395	0.397	0.434

[1]. The activate number of parameters at inference time is 39B.

Table 1: The *KPR* of various LLMs across different question categories. Overall performance on LONG^{2}RAG is reflected by ***Average***. **Highest** and second-highest *KPR* are highlighted for each question category.

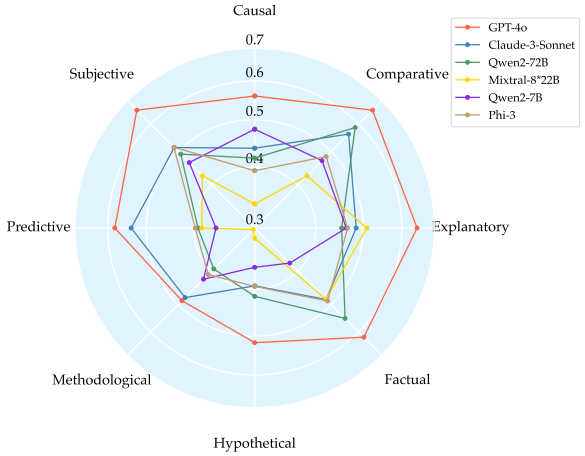


Figure 5: *KPR* of LLMs on different question categories.

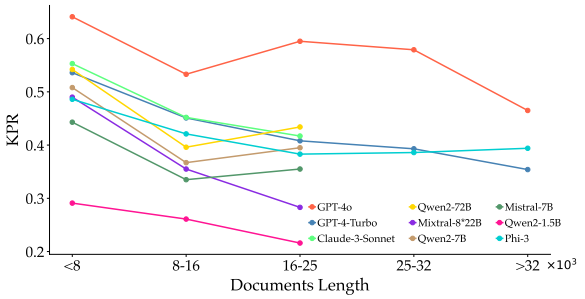


Figure 6: *KPR* of LLMs on different document lengths, where the horizontal coordinate is the length of the packed retrieved documents (in terms of tokens).

in the types of questions being considered, rather than the domains of expertise. It is noteworthy that we were pleasantly surprised to discover that nearly every model we evaluated performs exceptionally well on comparative questions, where the models adeptly incorporate important information from both sides of the comparison.

5.5 Results on Different Documents Length

To answer **RQ4**, we plot the correlation between input documents length after concatenation and *KPR* in Figure 6. Upon examining the figure, it is evident that there is a discernible trend: as the length of the input documents increases, the performance tends to deteriorate. This observation is consistent with recent research on long-context comprehension (Liu et al., 2024; Pal et al., 2023). Beyond the general trend, we have identified an intriguing pattern: a minor yet noticeable uptick in performance for several models when the generation length increases from 8-16K to 16-25K tokens. This performance rebound could potentially be attributed to the models' increased exposure to and familiarity with this range of data lengths during their training.

5.6 Results on Other Retrieval Strategy

Model	Trunk Size			Snippet	Summary	N/A
	512	1024	2048			
GPT-4o	0.549	0.568	0.557	0.403	0.342	0.579
Qwen2-72B	0.353	0.375	0.408	0.379	0.307	0.468

Table 2: The effect of different trunk sizes on LLMs' *KPR*. N/A: No truncation.

Remember that for a normal RAG pipeline, documents that are too long need to be truncated (Luo et al., 2024; Lyu et al., 2024). To answer **RQ5**, we first simulate the truncation process by cutting every document with a granularity of characters (Finardi et al., 2024), with the result as shown in Table 2. We also apply other processing methods for retrieved documents (Gao et al., 2023b) (*i.e.*, using GPT-4o to obtain the snippet and summary of

the retrieved documents). Note that we only use other approaches to process the document, while all other operations remain consistent. We observe that models that accept truncated inputs experience a decline in performance. Moreover, using the snippet of the document will result in a more pronounced performance degradation. The reason for this decline is quite straightforward: these strategies remove essential information from the source documents. Conversely, this observation corroborates the thought that *leveraging long-context to RAG pipelines, contributes to superior generation outcomes*. This is particularly evident when considering the perspective of information exploitation.

5.7 Results on Different Generation Length

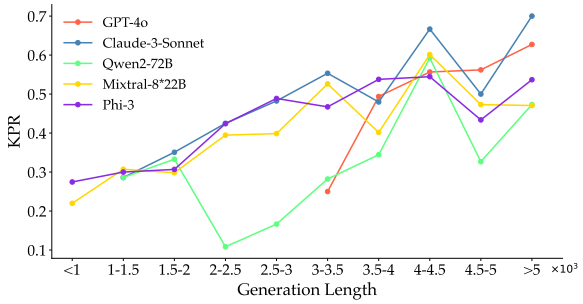


Figure 7: *KPR* of LLMs on different generation lengths (in terms of tokens).

We provide insight on **RQ6** by plotting the relationship between LLMs' generation length and *KPR*, as in Figure 7. The figure reveals that for certain models, performance is nearly proportional to the length of the generated content. However, an exception to this pattern is observed with Qwen2-72B, suggesting that merely promoting lengthy generation is not the sole determinant of performance.

5.8 Results of Different Evaluator

GPT-4o		**Llama3-70B**	
Model	KPR	Model	KPR
GPT-4o	0.579	GPT-4o	0.663
Claude3-Sonnet	0.477	GPT-4-Turbo	0.578
GPT-4-Turbo	0.469	Claude-3-Sonnet	0.568
Qwen2-72B	0.449	Qwen2-72B	0.562
Phi-3-mini-128K	0.434	Phi-3-mini-128K	0.559
Qwen2-7B	0.416	Qwen2-7B	0.544
Mixtral-8*22B	0.383	Mixtral-8*22B	0.505
Mistral-7B	0.373	Mistral-7B	0.490
Qwen2-1.5B	0.262	Qwen2-1.5B	0.342

Table 3: .The *KPR* for each model when using GPT-4o and Llama3-70B as evaluator

To answer **RQ7**, We use GPT-4o and Llama3-70b as evaluators and evaluate the *KPR* of GPT-4o and Phi-3-mini-128K with both evaluators. The results shown in Table 3 show that the model rankings remained largely consistent when evaluated by either GPT-4o or Llama3-70B, with only a minor position swap between Claude-3-Sonnet and GPT-4-Turbo. Llama3-70B generally assigned higher KPR scores due to its more lenient assessment of key points, while GPT-4o adopted a more conservative stance. These findings highlight the robustness of our comparative assessments.

6 Analysis

In this section, we provide a qualitative analysis of the annotation and an analysis of the evaluator.

6.1 Case Study on the Annotation

We conduct a deeper analysis of why the LLMs fail to include key points in its generation. As shown in Figure 8, we identify *three categories*: Incomplete expression, General expression, and Point missing. The incomplete expression category refers to instances where the generation partially includes the content of the key point but fails to cover it comprehensively. The general expression category typically occurs when a point is specific and detailed, yet the generation only vaguely or generally mentions it. The point missing category indicates cases where the generation completely omits a key point. In the first two categories, the generation still relatively incorporates information about the key point, whereas in the latter, the model fails to utilize key points to organize its generation. These three categories reflect the model's exploitation of key points during the generation process.

6.2 Accuracy of the Evaluator

To verify the reliability of GPT-4o as the $\text{LLM}_{\texttt{evaluate}}$, 180 cases are sampled, comprising model generation, key point, and the assessment of GPT-4o (binary: entail or not). These cases are drawn from the models we evaluate. Two annotators are then required to validate these sampled cases. We find GPT-4o achieves a pleasant average accuracy rate of 87%, demonstrating its reliability[7] as the evaluator. We report the IAA of the two annotators to be $\kappa = 0.61$ (Cohen's Kappa), indicating substantial agreement.

[7]This is also assured by setting the top-p temperature$= 0$ and fixed seed to ensure consistent evaluation results.

Incomplete expression:
Key point: Independent artists need to understand key music licensing terms such as Music Licensing, Music Lawyer, Synchronization License, Master Use License, and Rights Owners.
Generation: ... To use a piece of music in film or TV, two primary licenses are needed:
- Synchronization License: ...
- Master Use License: ...

General expression:
Key point: Types of analysis recommended for an income statement include benchmarks, comparisons over time, and non-GAAP metrics.
Generation: ... Analyze the income statement to understand the company's revenue, expenses, and net income. This will give you a snapshot of the company's profitability. ...

Point missing:
Key point: Developing legal AI software requires collaboration between AI experts and legal professionals.
Generation: ... Finally, a user-friendly interface is developed to allow lawyers to interact with the AI system. This interface should provide easy access to study materials, practice questions, and performance tracking tools. The goal is to create an intuitive platform that enhances the user's study experience. ...

Figure 8: Cases where the model generation fails to include a key point. The key point and the generation of LLMs are highlighted.

6.3 Quality of the retrieved documents

We conduct an additional experiment to assess document quality. Specifically, we truncate the documents used for key point extraction into segments of 1024 tokens each. For each segment, we compute the embeddings of both the question and the segment using the BAAI/bge large-en-v1.5 model, and then calculate the cosine similarity between these embeddings. The mean and standard deviation of the cosine similarity scores across all segments are 0.627 and 0.081, respectively. These results indicate that the documents utilized for extracting key points are of sufficiently high quality. The low standard deviation also suggests that the noises in those documents are low.

6.4 Additional Evaluation Result of Key Point Precision

Model	KPP	KPR	KPF	Response Length
GPT-4o	0.323	0.579	0.379	939.35
Phi-3-mini	0.337	0.434	0.354	832.93

Table 4: Evaluation results for key points in terms of precision and F1 score.

We incorporate two new metrics: Key Point Precision (*KPP*) and Key Point F1-score (*KPF*). *KPP* measures the proportion of key points in the generated response that are actually present in the retrieved documents. *KPF* combines *KPP* and Key Point Recall (*KPR*) to provide a more comprehensive evaluation of key point accuracy. We conduct an analysis of these metrics on 280 questions, comparing the performance of GPT-4o and Phi-3-mini-128K. The results are presented in Table 4 From the result, it can observed that while GPT-4o has a longer average response length, it has a lower *KPP* than Phi-3. Longer responses may not perform better across all metrics. The lower *KPP* for GPT-4o can be attributed to its tendency to incorporate parametric knowledge beyond the retrieved key points, demonstrating the depth of understanding but potentially reducing precision. The *KPR* score provides a balanced view of performance, considering both recall and precision. This helps mitigate the bias towards longer responses that might be present in *KPR* alone.

7 Conclusion

This paper introduces a novel benchmark, LONG^{2}RAG, and a corresponding evaluation metric, *KPR*, to address the limitations of existing benchmarks for evaluating long-context and long-form RAG in LLMs. LONG^{2}RAG features intricate and practical questions with associated retrieved documents that faithfully replicate the characteristics of real-world RAG scenarios. *KPR* focuses on evaluating the LLM's ability to effectively exploit the retrieved documents by measuring the recall of key points extracted from these documents within the generated response. We conduct an evaluation on 9 LLMs using LONG^{2}RAG and *KPR*, presenting novel insights and analysis.

8 Limitation

While the introduction of LONG^{2}RAG and *KPR* provides a significant step forward in evaluating long-context and long-form RAG, there are several limitations to this work. Firstly, the dataset, although diverse and carefully curated, is limited in size with only 280 questions. Expanding the dataset could provide more robust and generalizable insights. Secondly, the evaluation is primarily focused on English language content, which may not fully represent the capabilities of LLMs in handling other languages. Thirdly, the automated metric *KPR*, while innovative, may not perfectly capture the nuances of human evaluation, and its reliance on key point extraction could introduce additional biases or errors. Additionally, this study does not delve into the impact of different retrieval strategies on the performance of LLMs, which could be an important factor in RAG systems. Finally, the work primarily focuses on evaluating the effectiveness of LLMs in utilizing retrieved information, neglecting other aspects of long-form texts such as factual accuracy and logical coherence.

9 Ethics Statement

This work focuses on ethical considerations in developing and evaluating LLMs for RAG. We curated the LONG^{2}RAG benchmark to minimize data contamination and reflect real-world challenges like low signal-to-noise ratio and dispersed information in retrieved documents. We ensured the dataset does not contain personally identifiable information or offensive content. The proposed evaluation metric, *KPR*, assesses the effectiveness of LLMs in leveraging retrieved information, providing insights into how models exploit external knowledge. By promoting transparency and responsible evaluation practices, this research aims to contribute to the development of accurate and ethically sound LLMs.

Acknowledgements

The authors would like to thank the reviewers from the ACL Rolling Review for their thoughtful and constructive feedback. Their valuable insights have significantly enhanced the quality and clarity of our paper. This work was supported by National Key Research and Development Program of China (2023YFC3304800).

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005*, pages 65–72. Association for Computational Linguistics.

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.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2024. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762.

Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15607–15631, Toronto, Canada. Association for Computational Linguistics.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. 2023. Ragas: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*.

Alexander R Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2021. Qafacteval: Improved qa-based factual consistency evaluation for summarization. *arXiv preprint arXiv:2112.08542*.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. Eli5: Long form question answering. *arXiv preprint arXiv:1907.09190*.

Paulo Finardi, Leonardo Avila, Rodrigo Castaldoni, Pedro Gengo, Celio Larcher, Marcos Piau, Pablo Costa, and Vinicius Caridá. 2024. The chronicles of rag: The retriever, the chunk and the generator. *arXiv preprint arXiv:2401.07883*.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023a. Enabling large language models to generate text with citations. *arXiv preprint arXiv:2305.14627*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023b. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Shahriar Golchin and Mihai Surdeanu. 2023. Time travel in llms: Tracing data contamination in large language models. *arXiv preprint arXiv:2308.08493*.

Hangfeng He, Hongming Zhang, and Dan Roth. 2022. Rethinking with retrieval: Faithful large language model inference. *arXiv preprint arXiv:2301.00303*.

Aspen K Hopkins, Alex Renda, and Michael Carbin. 2023. Can llms generate random numbers? evaluating llm sampling in controlled domains. In *ICML 2023 Workshop: Sampling and Optimization in Discrete Space*.

Edward J Hu, Moksh Jain, Eric Elmoznino, Younesse Kaddar, Guillaume Lajoie, Yoshua Bengio, and Nikolay Malkin. 2023. Amortizing intractable inference in large language models. *arXiv preprint arXiv:2310.04363*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *CoRR*, abs/2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *CoRR*, abs/2401.04088.

Kalpesh Krishna, Yapei Chang, John Wieting, and Mohit Iyyer. 2022. RankGen: Improving text generation with large ranking models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 199–232, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4940–4957, Online. Association for Computational Linguistics.

Yucheng Li. 2023. An open source data contamination report for llama series models. *arXiv preprint arXiv:2310.17589*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-eval: Nlg evaluation using gpt-4 with better human alignment (2023). *URL http://arxiv. org/abs/2303.16634*.

Kun Luo, Zheng Liu, Shitao Xiao, and Kang Liu. 2024. Bge landmark embedding: A chunking-free embedding method for retrieval augmented long-context large language models. *arXiv preprint arXiv:2402.11573*.

Yuanjie Lyu, Zhiyu Li, Simin Niu, Feiyu Xiong, Bo Tang, Wenjin Wang, Hao Wu, Huanyong Liu, Tong Xu, and Enhong Chen. 2024. Crud-rag: A comprehensive chinese benchmark for retrieval-augmented generation of large language models. *arXiv preprint arXiv:2401.17043*.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore. Association for Computational Linguistics.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Arka Pal, Deep Karkhanis, Manley Roberts, Samuel Dooley, Arvind Sundararajan, and Siddartha Naidu. 2023. Giraffe: Adventures in expanding context lengths in llms. *arXiv preprint arXiv:2308.10882*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Justus J Randolph. 2005. Free-marginal multirater kappa (multirater k [free]): An alternative to fleiss' fixed-marginal multirater kappa. *Online submission*.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. 2023. Ares: An automated evaluation framework for retrieval-augmented generation systems. *arXiv preprint arXiv:2311.09476*.

Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. 2023. Zeroscrolls: A zero-shot benchmark for long text understanding. *arXiv preprint arXiv:2305.14196*.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: Factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8273–8288, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Alessandro Stolfo. 2024. Groundedness in retrieval-augmented long-form generation: An empirical study. *arXiv preprint arXiv:2404.07060*.

Haochen Tan, Zhijiang Guo, Zhan Shi, Lu Xu, Zhili Liu, Xiaoguang Li, Yasheng Wang, Lifeng Shang, Qun Liu, and Linqi Song. 2024. Proxyqa: An alternative framework for evaluating long-form text generation with large language models. *arXiv preprint arXiv:2401.15042*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Jerry Wei, Chengrun Yang, Xinying Song, Yifeng Lu, Nathan Hu, Dustin Tran, Daiyi Peng, Ruibo Liu, Da Huang, Cosmo Du, et al. 2024. Long-form factuality in large language models. *arXiv preprint arXiv:2403.18802*.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. 2023. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023a. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

Fangyuan Xu, Yixiao Song, Mohit Iyyer, and Eunsol Choi. 2023b. A critical evaluation of evaluations for long-form question answering. *arXiv preprint arXiv:2305.18201*.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2023c. Retrieval meets long context large language models. *arXiv preprint arXiv:2310.03025*.

Rongwu Xu, Xuan Qi, Zehan Qi, Wei Xu, and Zhijiang Guo. 2024a. Debateqa: Evaluating question answering on debatable knowledge. *arXiv preprint arXiv:2408.01419*.

Rongwu Xu, Zehan Qi, Cunxiang Wang, Hongru Wang, Yue Zhang, and Wei Xu. 2024b. Knowledge conflicts for llms: A survey. *arXiv preprint arXiv:2403.08319*.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 27263–27277.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yifan Zhang, Yifan Luo, Yang Yuan, and Andrew C Yao. 2024a. Autonomous data selection with language models for mathematical texts. In *ICLR 2024 Workshop on Navigating and Addressing Data Problems for Foundation Models*.

Zihan Zhang, Meng Fang, and Ling Chen. 2024b. Retrievalqa: Assessing adaptive retrieval-augmented generation for short-form open-domain question answering. *arXiv preprint arXiv:2402.16457*.

A Details about LONG^{2}RAG

A.1 Distribution of LONG^{2}RAG

We show the domain distribution and question category distribution of the questions in LONG^{2}RAG in Figure 10 and Figure 11, respectively. It can be seen that both distributions are relatively uniform. We show one instance of each domain in Figure 9. Moreover, the length distribution of all retrieved documents is depicted in Figure 12.

A.2 The Score Function from AutoDS

In an effort to streamline the process of pre-filtering key points extracted by LLMs that do not adhere to our criteria, thereby reducing the workload of human annotators, we have implemented the scoring function available in AutoDS (Zhang et al., 2024a). The rationale behind this approach is that LLMs often struggle with quantitative evaluation, particularly when it comes to assigning numerical scores that accurately reflect the importance of the extracted information (Hopkins et al., 2023; Hu et al., 2023). Nevertheless, DPO (Rafailov et al., 2024) demonstrated that logits can be employed as a score function, and AutoDS also adopts this strategy.

This function evaluates the LLM's propensity to agree or negate a claim in the content. With a carefully designed prompt, this function operates on the logits corresponding to "YES" and "NO" responses to achieve content evaluation. The equation of the

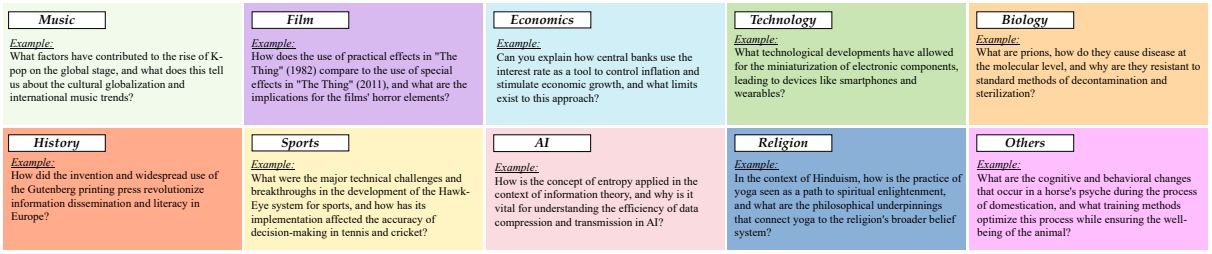


Figure 9: Examples of questions for each domain.

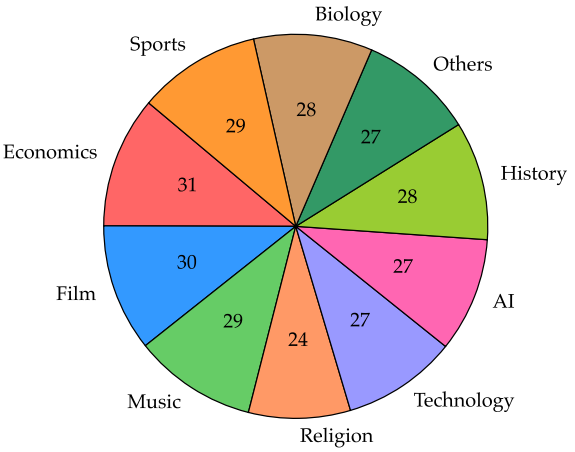


Figure 10: Domain distribution of questions in LONG^{2}RAG.

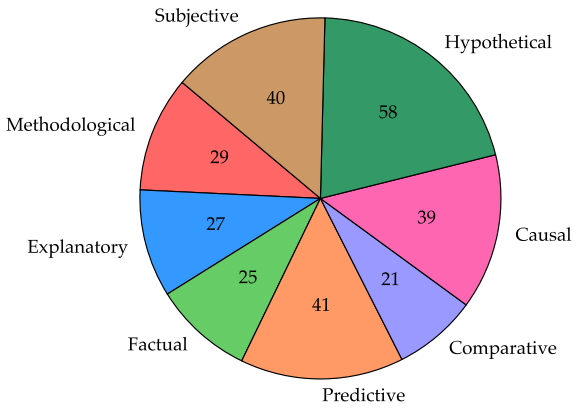


Figure 11: Question category distribution of questions in LONG^{2}RAG.

score function is:

$$\text{Score}(\cdot) = \frac{\exp(\text{logit}(\text{“YES”}))}{\exp(\text{logit}(\text{“YES”})) + \exp(\text{logit}(\text{“NO”}))} \quad (3)$$

In our work, we employ LLama3-70b-Instruct to compute the logits in Equation 3, utilizing the following prompt:

Prompt for filtering the key points

```
You are a helpful AI assistant. Your role is to evaluate whether a piece of information can serve as a key point in answering the question.
question: {question}
information: {point}
Can the above information directly help in addressing the question (thus being a key point)? You must respond with YES or NO.
```

The score distribution of all the 28,611 points extracted is shown in Figure 13. The fact that we can observe a near-diagonal distribution from the CDF (Cumulative Distribution Function) of Figure 13 suggests Equation 3 is well calibrated. To determine an appropriate threshold, we randomly select 20 questions accompanied by 1,051 points and conduct annotation by two annotators. We compute the false positive rate for various threshold scores, which indicates the proportion of erroneously identified non-key points among the filtered points. The false positive rate of each individual's annotation results is averaged. We opt for a threshold of 0.7 due to its associated false positive rate being 8.7%, thereby facilitating substantial data filtration while maintaining a tolerable error margin.

A.3 Human Verification Details

Four annotators are recruited to verify the key points extracted by the LLM. All the annotators are college students who major in English. The interface of the provided annotation tool is shown in Figure 14 alongside the annotation manual in Figure 15, Figure 16, Figure 17 and Figure 18.

This annotation interface contains the question, the retrieved document, and the extracted key point. The annotators are required to complete two tasks, including:

- **Task 1 "Can this Point be found in the Text?"**: Verifying whether this point is grounded by the document.
- **Task 2 "Whether this Point is a Key Point"**:

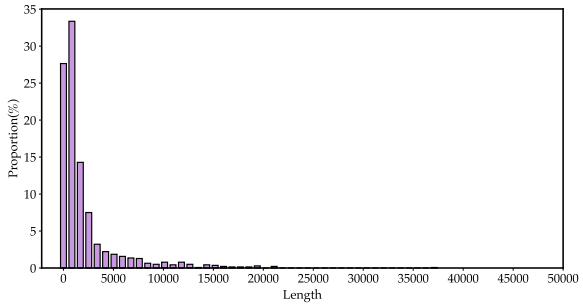


Figure 12: Length distribution of all retrieved documents. The horizontal coordinate represents the number of words in each document, while the vertical coordinate denotes the proportion.

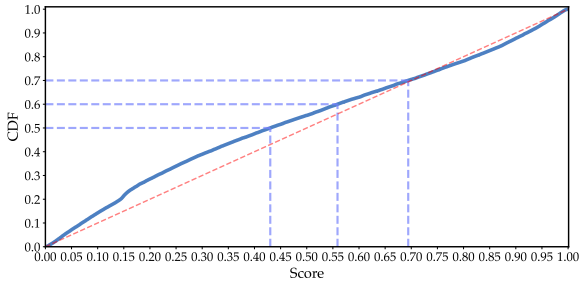


Figure 13: Score distribution of all points. The three dot lines in the plot indicate the scores when the CDF is 0.5, 0.6, and 0.7, respectively. The red dot line is a $y = x$ diagonal. The closer to the red line, the better the calibration.

> Verifying whether this point is a key point. Three options are provided where "Agree" signifies the ability to directly address the question, "Neutral" indicates the provision of relevant background information related to the question without directly answering it, and "Disagree" denotes content that is not pertinent to the question and should be omitted.

We evenly distributed the key points derived from 560 questions among four annotators, with each annotator responsible for annotating key points associated with 140 questions.

To ensure annotation reliability, we implemented a stringent quality control process. For this process, 10% of the key points were annotated by all four annotators. This overlapping subset served as a control group, allowing us to verify the consistency and quality of annotations across individuals.

Due to the quality constraints, we only selected the annotated data from two annotators to form our dataset, LONG^{2}RAG, which comprises 280 questions. From this dataset, we randomly sampled 200 key points that were jointly annotated by these two annotators. To evaluate the inter-annotator agreement, particularly in light of category imbalances, we utilized the free-marginal multi-rater kappa statistic (Randolph, 2005). The kappa value of 0.56 indicates a moderate level of agreement among the annotators.

B Prompts Employed in Dataset Construction

In the procedure of constructing LONG^{2}RAG with the automatic pipeline utilizing $\text{LLM}_{\texttt{dataset}}$, we employ a series of prompts. Readers are expected to generate additional dataset samples in accordance with the guidelines outlined in § 4 and by applying the following prompts for future research.

Prompt for filtering questions in ELI5: `Please evaluate and score a given query based on the following criteria. Each satisfied criterion earns one point, with a maximum score of 5 points:`
`1. Exclude common sense questions: Filter out any queries that can be answered with basic common knowledge or a simple search engine query. Example: "What is the boiling point of water?" or "What is the highest mountain in the world?"`
`2. Ensure clarity: Filter out any queries that are unclear or have a vague scope. Example: "Tell me something interesting." or "Explain this."`
`3. Ensure no ambiguity: Filter out any queries that could have multiple interpretations, ensuring the query has one clear answer or direction. Example: "How do you do this?" (needs to specify what "this" refers to)`
`4. Complexity requirement: Select queries that require deep thinking, detailed explanation, or multi-step reasoning. Example: "How can graph neural networks be used to optimize recommendation algorithms in social networks?"`
`5. Subjective opinion: Select queries that require the user to provide personal insights or subjective opinions. Example: "What are your thoughts on the future of artificial intelligence in healthcare?"`
`Please score the given query based on these criteria, with a range of 1 to 5 points. You should output the score wrapped with [], like [score].`
`Here is the query:`
`query`
`Your score is:`

Prompt for in-width evolving: `You are a question generator. Your goal is to draw inspiration from the #Given Question# to create a brand new question.`

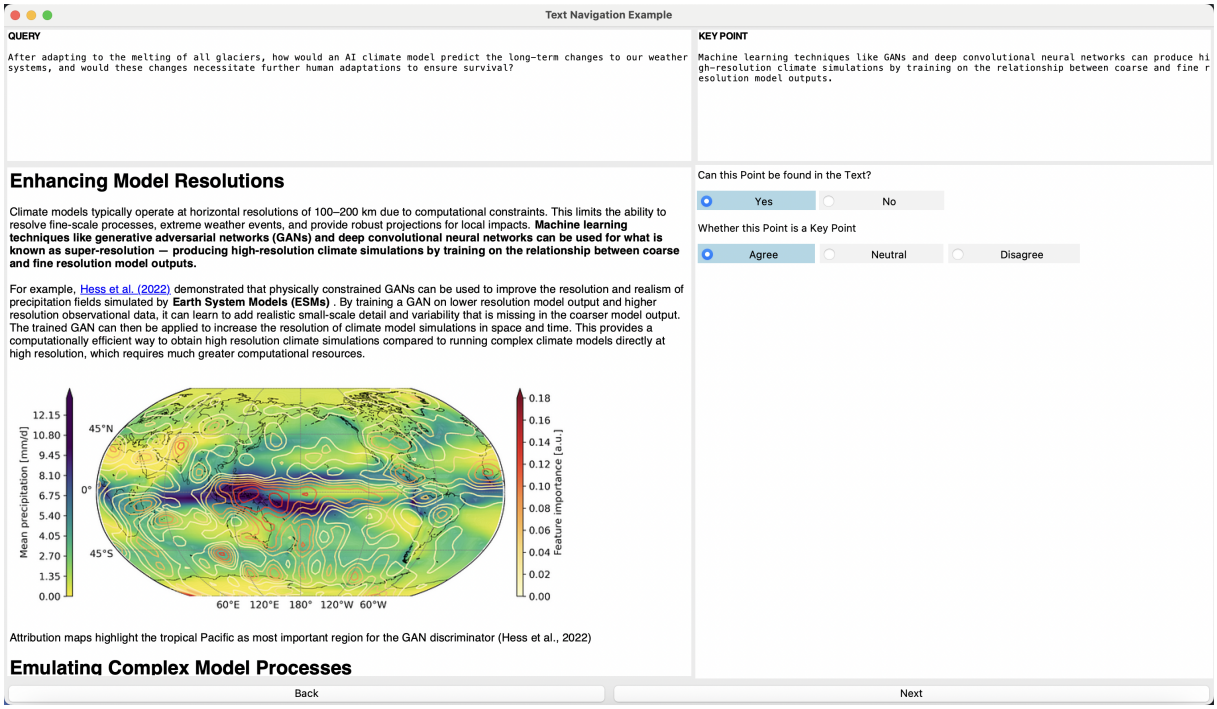


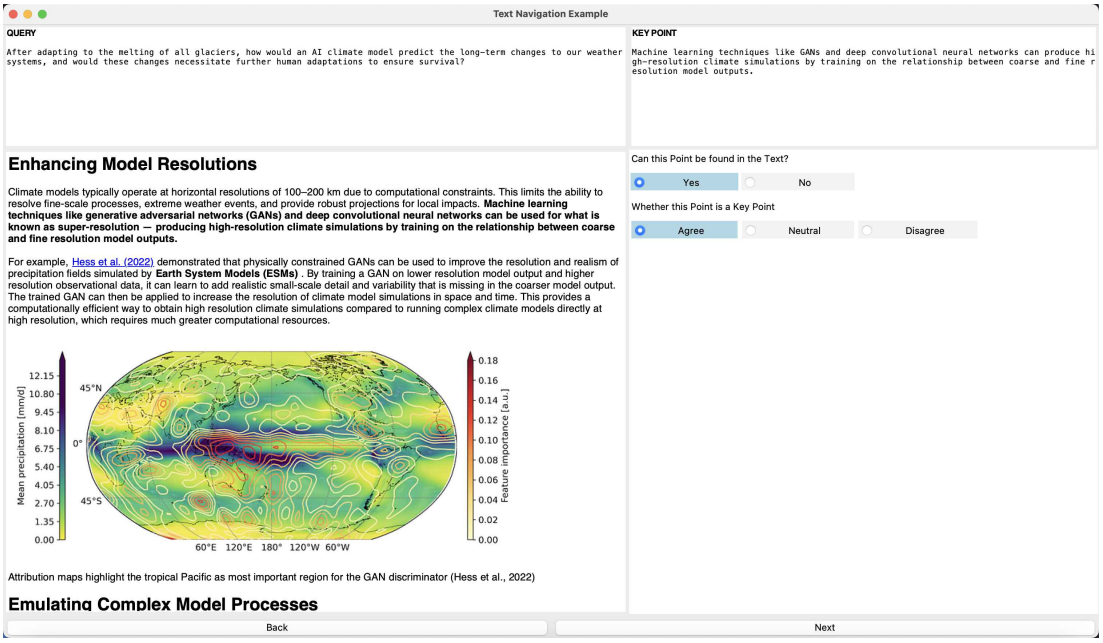
Figure 14: Our annotation interface. The interface comprises four utility areas. The upper left area displays the question, the lower left area presents the retrieved document, the upper right area represents the key point, and the lower right area is used for annotation.

This new question should belong to the same question category as the #Given Question#. But remember, the #Created Question# should be closely related to {topic} topic.
The LENGTH and difficulty level of the #Created Question# should be similar to that of the #Given Question#.
The #Created Question# must meet the following requirements:
1. The question should require a long-form response that includes several specific details.
2. Do not generate commonsense, vague, or ambiguous questions.
3. The question should be reasonable and can be responded to by humans.
4. You are encouraged to generate questions that require deep thinking, detailed explanations, or multi-step reasoning.
5. The #Created Question# should be very specific and niche within the topic of {topic}. The created question must be closely related to the topic of {topic}.
6. Follow the question styles in the #Given Question#. The #Given Question# belongs to type {question_type}.
7. Wrap the #Created Question# in square brackets, like [created question]. You can generate multiple questions once a time and wrap each question in square brackets, like created question i: [question i].
#Given Question#:
{seed_question}
#Created Question#:

Prompt for in-depth evolving: You are a question generator. Your goal is to draw inspiration from the #Given Question# to create a brand new question.
This new question should become the more complex version of the #Given Question# to make those famous AI systems (e.g., ChatGPT and GPT4) a bit harder to handle. But remember, the #Created Question# should be closely related to {topic} topic.
You can complicate the given prompt using the following methods: deepening, concretizing, increasing reasoning steps, and complicating input.
You should try your best not to make the #Created Question# become verbose, #Created Question# can only add 10 to 20 words into #Given Question#.
This new question should belong to the same question category as the #Given Question#.
The #Created Question# must meet the following requirements:
1. The question should require a long-form

Annotation Manual

1. System introduction



The system consists of four main sections

- Board 1: Query, that is, the question
- Board 2: Text, the document that the system refers to in order to answer the question.
- Board 3: Key Point, the point extracted from the document that facilitates the answer to the question.
- Plate 4: The part to be annotated, by selecting the corresponding option to complete the annotation. There are two annotation tasks to be completed
    - Whether the Key Point in Plate 3 can be found in the Text in Plate 2 (we have highlighted the reference point to help determine this, but you need to be aware that this highlighting may not be 100% accurate, i.e. you should browse the rest of the document if the highlighting is not accurate).
    - Whether the Key Point in Board 3 is a Key Point relative to Query in Board 1

Figure 15: Annotation manual (Page 1 of 4).

2. Annotation Requirements

This labeling task requires the completion of two labeling tasks, both listed in board 4

1. **Can this point be found in the Text**

Can this point be found in the Text in Board 3.

Sentences that may be relevant to the point are highlighted in the documentation for board 2 to help with labeling.

The following steps should be taken by the annotator to annotate:

- Determine whether the bullet point in board 3 can be extracted from the document in board 2 based on the highlighted snippet
- If not, it is necessary to read the entire content of the document in plate 2 to make a judgment, not entirely based on the highlighted fragment as a basis for judgment.

There are two options, Yes and No, corresponding to the following criteria:

- Yes: the points in board 3 can be extracted from the documents in board 2
- No: The points in board 3 cannot be extracted from the documents in board 2.

2. **Whether the point is a utility point**

If in the first step it was confirmed that the point in board 3 cannot be extracted from the document in board 2, i.e. the option is No, then this step can be skipped. If the option is Yes, then this step is required.

In this annotation step, two conditions need to be judged:

- Whether or not the points in board 3 can help answer the questions in board 1
- whether the points in Plate 3 are key points

Based on the judgment of these two conditions, there are three options as follows:

- Agree: being able to answer the question directly.
- Neutral: Relevant to the question but not able to answer it, e.g. background information. Can appear in the answer, but is not required
- Disagree: Should not be in the answer, not relevant to the question, cannot answer the question

Figure 16: Annotation manual (Page 2 of 4).

3. Operational issues

- At the bottom there are two buttons, Back and Next, which can be clicked to switch between the next and previous question. Note that you can only click next if you have selected both options, so if the first option is No, the second one can be selected at random.
- For the options in board 4, every time you choose, the corresponding results will be saved to the choices.jsonl file in the data folder in the background, so the annotator doesn't need to click on the save, it just needs to click on the options to choose.
- For part of the labeled data, due to the large number of words in the text, the loading time is long, so when clicking Back and Next to switch, you need to wait for a number of seconds.
- If the system crashes, or needs to be shut down in the middle. The user can shut down the system. When restarted again, the previous operation of the option will be automatically restored. The user needs to click on the bottom of the jump to switch to the last labeling questions, jump button on the left side of the input box, directly enter the corresponding question number can be, the question number appears in the upper right corner next to the key point

4. Demonstrations

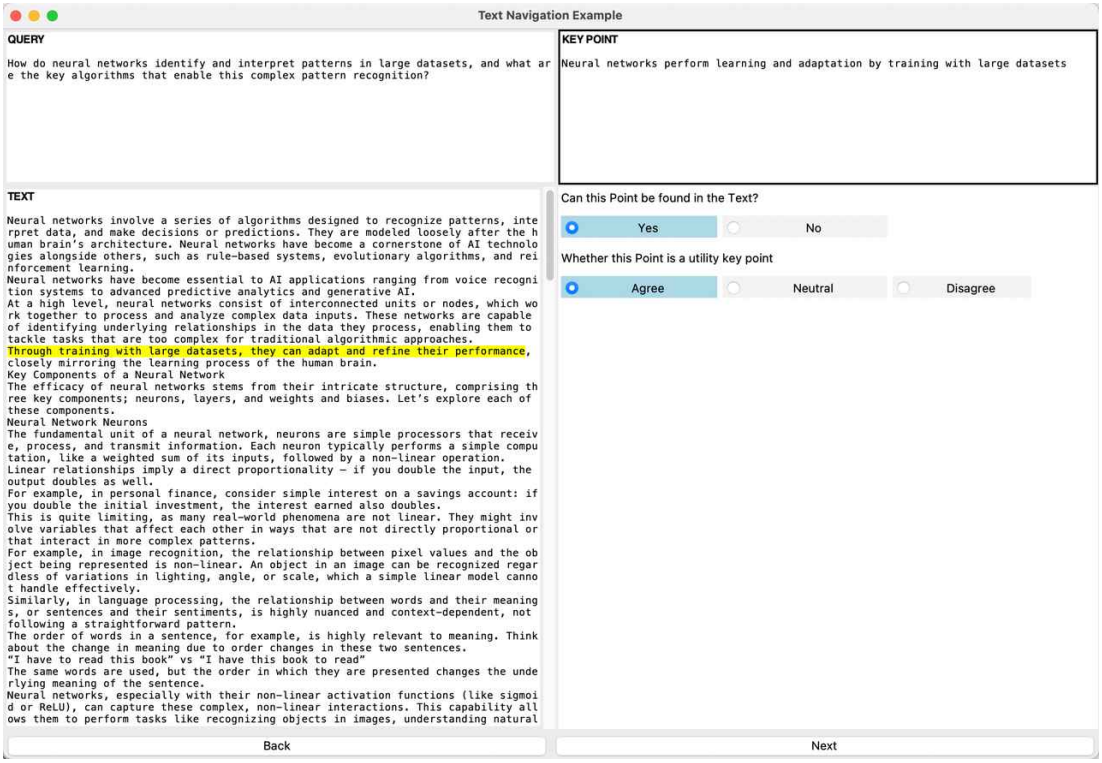
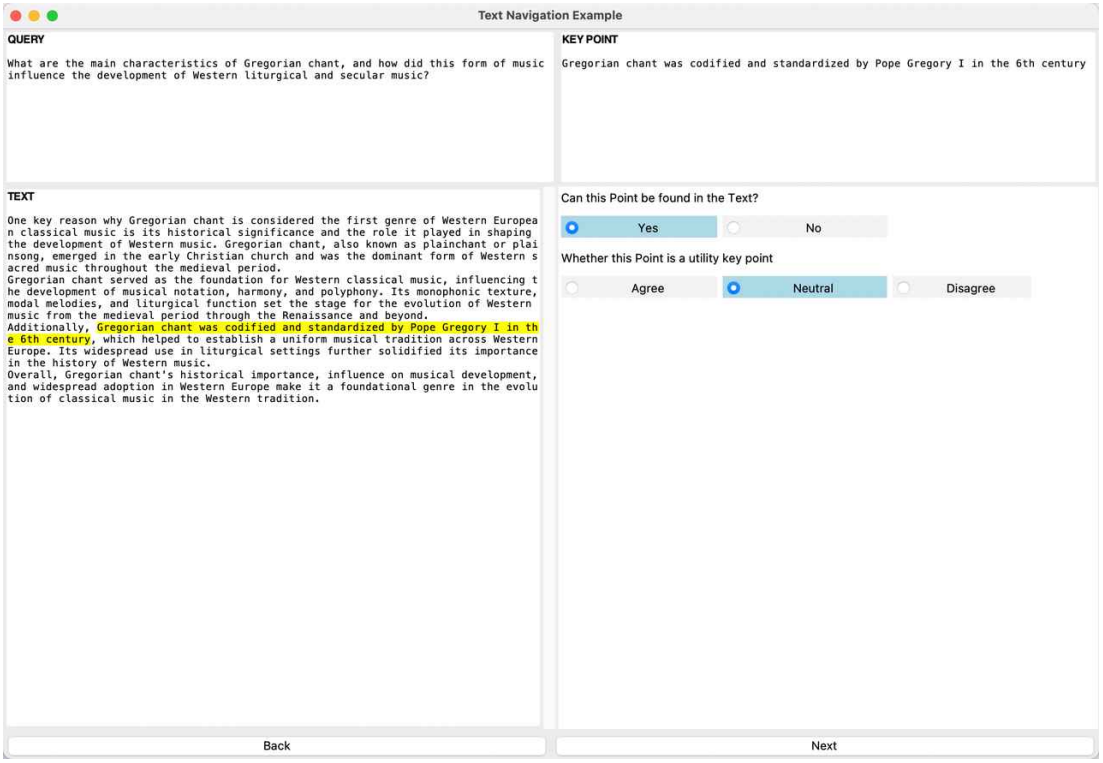


Figure 17: Annotation manual (Page 3 of 4).



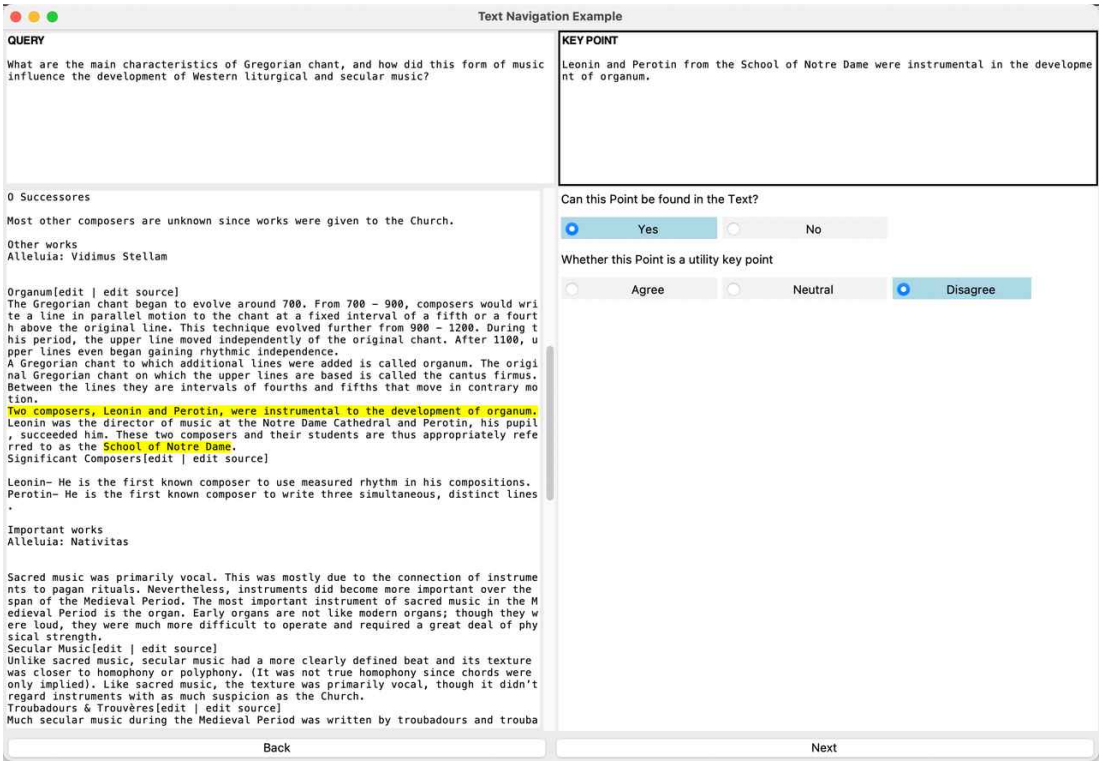


Figure 18: Annotation manual (Page 4 of 4).

response that includes several specific details.
2. Do not generate commonsense, vague, or ambiguous questions.
3. The question should be reasonable and can be responded to by humans.
4. You are encouraged to generate questions that require deep thinking, detailed explanations, or multi-step reasoning.
5. The #Created Question# should be very specific and niche within the topic of {topic}. The created question must be closely related to the topic of {topic}.
6. Follow the question styles in the #Given Question#. The #Given Question# belongs to type {question_type}.
7. Wrap the #Created Question# in square brackets, like [created question]. You can generate multiple questions once a time and wrap each question in square brackets, like created question i: [question i].
#Given Question#:
{seed_question}
#Created Question#:

Prompt for decomposing questions for retrieval:
You are a language analysis assistant capable of analyzing user questions to determine if and how to decompose the question.
Problem Definition
1. Simple question: Direct questions that only require simple information and do not include metaphors, multi-hop, multi-entity, or other complex logic questions;
2. Complex question: Includes multi-hop, metaphors, multi-entity, complex conditions, etc., requiring in-depth analysis and thought, as well as support from various types of information;
3. Vague question: Refers to questions with unclear intent, lacking a query subject, or ambiguously expressed content.
Task Requirements
1. For simple questions, there is no need to decompose the question;
2. For complex questions, decompose the complex question into multiple sub-questions based on the specific content of the question, ensuring the sub-question has a complete intent. Sub-questions need to be more concise and easier to search;
3. For vague questions, reasonably expand based on the information provided in the question, generating multiple related questions, each covering different aspects of the query subject.
Analyze user questions according to the given requirements and provide results. The results should be output in the format of an inline JSON.
Output format:
"'json
"Question Type": "Simple/Complex/Vague", "Sub-questions": ["..."]
"'
Question: {query}

Prompt for extracting key points: Based on the text provided, identify key points in the text that directly help in responding to the query.
Format your response as follows: each point should start with "Point [number]:", followed by its content and spans in the text that entails the key point.
IMPORTANT: The output must be of the format " Point [number]: <point_start>[content of point] <point_end><span_start>[span1]<span_end><span_start>[span2]<span_end>"
IMPORTANT: Ensure each point is helpful in responding to the query. Keep the point using the original language and do not add explanations.
IMPORTANT: Each span must be a single consecutive verbatim span from the corresponding passages. Copy verbatim the spans, don't modify any word!
Here is an example:
<One-shot Demonstration>
Remember:
- key points can be abstracted or summarized, but the span must be a copy of the original text. The content of the key point does NOT need to be the same as that of the span.
- These key points must be helpful in responding to the query.
- Copy verbatim the spans, don't modify any word! If there are multiple spans for a point, separate them with <span_start> and <span_end> tokens.
[Query]: {query}
[Text]: {text}
[Key Point]

Prompt for entailment identification: Your task is to determine whether a claim is entailed with a document.
You should evaluate whether the information in the document supports or describes the claim provided. You must provide an answer based on whether the document does entail the claim, does not entail the claim, or is neutral.
If the claim is entailed, you should provide snippets from the document that support this claim.
Document: {document}
Claim: {claim}

Provide your answer as [yes], [no], or [neutral]. State your reason behind the answer and provide snippets from the document that support this claim if your answer is [yes].

Prompt for key point deduplication and aggregation: Based on the points extracted from a piece of text, identify and remove any duplicate points to streamline the list.

REMEMBER:

- The de-duplicated points need to contain all the original points.
- An original point cannot exist in two different de-duplicated points at the same time.
- Format your response as follows: each unique point should start with "Point [number]:", followed by its content and corresponding original point numbers. The corresponding original point numbers should be wrapped with [].

Here is an example:

<One-shot Demonstration>

Remember to not add any new points, only de-duplicate the existing ones. And do not add any explanations.

[Original Points]

{points}

[De-duplicate Points]