Summary of a Haystack:
A Challenge to Long-Context LLMs and RAG Systems

Philippe Laban[*]   **Alexander R. Fabbri**[*]   **Caiming Xiong**   **Chien-Sheng Wu**

Salesforce AI Research

{plaban, afabbri, cxiong, wu.jason}@salesforce.com

Abstract

LLMs and RAG systems are now capable of handling millions of input tokens or more. However, evaluating the output quality of such systems on long-context tasks remains challenging, as tasks like Needle-in-a-Haystack lack complexity. In this work, we argue that summarization can play a central role in such evaluation. We design a procedure to synthesize Haystacks of documents, ensuring that specific *insights* repeat across documents. The "Summary of a Haystack" (SummHay) task then requires a system to process the Haystack and generate, given a query, a summary that identifies the relevant insights and precisely cites the source documents. Since we have precise knowledge of what insights should appear in a haystack summary and what documents should be cited, we implement a highly reproducible automatic evaluation that can score summaries on two aspects – Coverage and Citation. We generate Haystacks in two domains (conversation, news), and perform a large-scale evaluation of 14 LLMs and corresponding 50 RAG systems. Our findings indicate that SummHay is an open challenge for current systems, as even systems provided with an Oracle signal of document relevance lag our estimate of human performance (56%) by 10+ points on a Joint Score. Without a retriever, long-context LLMs like GPT-4o and Claude 3 Opus score below 20% on SummHay. We show SummHay can also be used to study enterprise RAG systems and position bias in long-context models. We hope future systems can equal and surpass human performance on SummHay.

1 Introduction

Recent progress in efficient attention mechanisms has led to the expansion of the context length of large language models (Beltagy et al., 2020; Su et al., 2024). Previous state-of-the-art models such as T5 (Raffel et al., 2020) and BART (Lewis et al., 2019) were limited to input contexts of 512 or 1024 tokens, while the latest models such as Claude-3 or Gemini-1.5-pro (Reid et al., 2024) can process sequences of hundreds of thousands or millions of tokens.

Another paradigm, Retrieval Augmented Generation (RAG) (Lewis et al., 2020; Guu et al., 2020), has emerged as an alternative to these long-context LLMs, proposing a pipelined approach in which a retriever dynamically selects the context relevant to a given input query, alleviating the need for the generator to process long contexts directly.

Although both RAG and long-context LLMs offer to solve the common problem of answering queries over a large corpus of text, a direct comparison on a common task is still lacking, and evaluating such systems is an open challenge. Some recent work has popularized tests such as the Needle-in-a-Haystack task (Kamradt, 2023), which requires models to identify a small piece of information in a large document. However, these tasks do not offer the complexity needed to differentiate the capabilities of the latest generation of large-language models, with several state-of-the-art models achieving near-perfect performance.

In this work, we propose to leverage the task of summarization as a testbed for evaluating long-context models and RAG systems. Summarization requires reasoning over a long context and a careful understanding of the relative importance of content. However, most prior work on summarization evaluation, particularly in evaluating the relevance of summaries, has focused on single-document summarization or tasks in which the input content is on the order of 1,000-2,000 tokens (Laban et al., 2020; Fabbri et al., 2021a; Bhandari et al., 2020; Liu et al., 2022). Apart from Chang et al. (2023), which focuses on summary coherence for 100k-token books, other evaluation work on longer conversational and multi-document news summarization is still often limited to around 10k tokens (Zhong et al., 2021;

---

[*] Equal contribution

Huang et al., 2023).

A central problem is that summarization evaluation often relies on low-quality reference summaries and automatic metrics that do not correlate well with human judgments. Within reference-based evaluation, a candidate summary is compared to a gold-standard reference summary, with the optics that a higher overlap between the candidate and reference summary indicates higher quality. This paradigm may limit evaluation reliability, due to the lack of gold-standard references, particularly in long-context settings where obtaining high-quality summaries would be prohibitively expensive. Furthermore, automatic metrics may still fail to correlate well with human judgments with respect to these references; despite the human-validated pipeline of Huang et al. (2023), the best automatic metric for content coverage in that study has a correlation of just 0.37 with human judgment.

In this work, we address these limitations through synthetic data generation. An overview of our Framework is found in Figure 1. We propose data synthesis programs to generate a large corpus of documents (the "Haystack") on a given topic. By enforcing that certain units of information ("insights"), categorized according to various subtopics, repeat within Haystack documents, and precisely controlling which insights occur in which documents, we can automatically derive the relevant insights within the Haystack for a given search query. A system completing the Summary of a Haystack (SummHay) task must then summarize insights relevant to a search query and cite the source documents of each insight. These summaries can be evaluated based on whether they cover the expected reference insights, and cite precisely and thoroughly the source documents.

Our first contribution is a procedure for generating Haystacks in two domains: conversations and news articles. Section 3 details the carefully designed pipeline to ensure the feasibility and validity of the task. A Haystack typically contains 100 documents on a topic, totaling approximately 100k tokens. We generate a total of 10 Haystacks, each coupled with roughly 10 queries, for a total of 92 SummHay tasks. Our pipeline can be scaled and applied to other domains.

Our second contribution develops SummHay's evaluation protocol, centering on evaluating system outputs on their Coverage of reference insights, and the quality of their Citation. A manual annotation confirms the strong reproducibility of the

protocol among knowledgeable annotators (0.77 correlation). We then experiment with LLM-based evaluation, finding that although the level of correlation is slightly lower (0.71), evaluation cost is reduced by a factor of almost 50.

Our third contribution is an estimate of human performance on SummHay and a large-scale evaluation of 50 RAG systems and 14 long-context LLMs. Our findings indicate that: (1) SummHay is challenging for all systems we evaluate, with all models significantly below our estimate of human performance, even when given oracle signals of document relevance; (2) non-trivial trade-offs exist when choosing between a RAG pipeline and a long-context LLM, with RAG systems typically improving citation quality, at the cost of insight coverage, (3) using advanced RAG components (e.g., Cohere's Rerank3) leads to end-to-end performance boosts on the task, confirming that SummHay is a viable option for holistic RAG evaluation, (4) a positional bias experiment on SummHay confirms the *lost in the middle* phenomenon, demonstrating that most LLMs are biased towards information at the top or bottom of the context window.

Our code and dataset are publicly released to allow for the extension of the work[1]. A system that achieves a high score on SummHay can reliably reason over large corpora of documents, detect and summarize insights, and accurately cite its sources. Although human performance on SummHay is still out of reach, we anticipate future systems will achieve and surpass such performance, providing more reliable and trustworthy answer engines.

2 Related Work

2.1 Summarization Evaluation

Existing work in summarization relevance, or coverage, evaluation has largely focused on the short-input, single-document setting (Gao and Wan, 2022; Fabbri et al., 2021b). Extending to long input evaluation, recent work has performed meta-evaluation on coherence in book summarization (Chang et al., 2023) and faithfulness across several domains (Krishna et al., 2023; Min et al., 2023; Zhang et al., 2024a). For coverage evaluation, recent work has studied content selection for book summarization (Kim et al., 2024), evaluated a two-step extract-evaluate framework (Wu et al., 2023), and compared the correlation of LLM metrics in

---

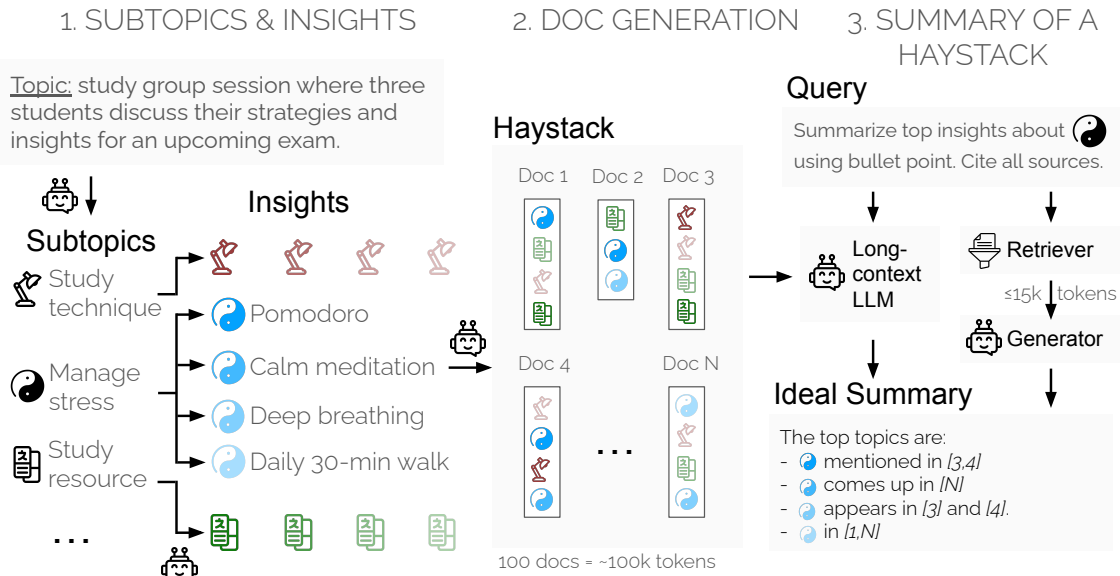[1] https://github.com/salesforce/summary-of-a-haystack

Figure 1: Diagram illustrating the steps to synthesize a Haystack of documents given an input scenario: subtopic and insight creation followed by document generation. Once a Haystack is synthesized, it can be used to benchmark LLMs / RAG systems on query-focused summarization tasks.

coverage (Huang et al., 2023). We leverage summarization relevance as a test-bed for long-context evaluation, and we focus on our synthetic creation pipeline and the simplified relevance evaluation that results in a high-correlation automatic metric.

2.2 Long-Context LLM Evaluation

Needle-in-a-haystack (Kamradt, 2023) was proposed to assess the long-context recall ability of LLMs. Subsequent work has analyzed the effect of needle placement (Machlab and Battle, 2024) and multi-needle (LangChain, 2024) and multi-modal variations (Song et al., 2024; Reid et al., 2024).

Additionally, several long-context evaluation benchmarks have been created, for example by building upon and revising existing tasks and datasets (Bai et al., 2023; An et al., 2023) Some work proposes ways to extend the context length of shorter-context datasets; (Kuratov et al., 2024; Kwan et al., 2023), while other work addresses data contamination in long-context settings (Ni et al., 2024; Dong et al., 2023). Several papers introduce synthetic data in addition to existing tasks (Shaham et al., 2023; Zhang et al., 2024b), which can prove to be more difficult for current models, as seen in Bai et al. (2023). Our benchmark focuses on synthetic data on the scale of 100k input tokens, and as opposed to existing synthetic tasks centered largely around retrieving, counting, or sorting, our summarization task requires aggregating and non-trivial reasoning over the long context.

2.3 Attribution Evaluation

Several benchmarks have emerged to study the ability of LLMs to ground their generation with citations (Li et al., 2023a). AttributionBench (Li et al., 2024) aggregates 7 existing attribution datasets, including Hagrid (Kamalloo et al., 2023), consisting of generative answers to questions annotated by humans for attribution, and AttrEval-GenSearch (Yue et al., 2023), which categorizes attribution into three levels of support. Attribution evaluation has also been performed along sources beyond documents such as knowledge graphs (Hu et al., 2024; Li et al., 2023b) and for tasks such as long-form question-answering (Chen et al., 2023b). Specific to summarization, Seahorse (Clark et al., 2023) collects annotations for summary attribution in the short-context setting. In this paper, we study attribution, or citation, in the context of long-input summarization. Due to our synthetically generated data, we can trace reference insights to their sources and directly evaluate summary citations.

3 Summary in a Haystack Framework

Figure 1 illustrates the process of synthesizing Haystack data, and the task, which we now detail.

3.1 Preliminaries

In SummHay, as in the needle-in-a-haystack task, the LLM responds to a query, but here it must generate a long-form answer (200-300 words) that requires identifying and summarizing insights that re-

3 REFERENCE INSIGHTS

One student suggests taking a 5-minute break after every 25 minutes of studying, and mentions the **Pomodoro technique** as helpful.
Reference Docs: 8, 32, 79, 83, 95
<u>Coverage: 100;</u> Citation P: 50; R: 20; <u>F1: 29</u>

A student recommends using a specific **meditation app called 'Calm'** that they use for 15 minutes each morning to manage stress.
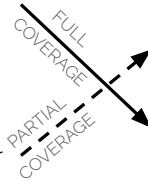*Reference Docs: 11, 30, 46, 53, 79, 80*
<u>Coverage: 50;</u> Citation P: 80; R: 67; <u>F1: 73</u>

One student shares that they do 10 minutes of **deep breathing** exercises each night before going to bed to help reduce stress.
Reference Docs: 8, 32, 46, 53, 69, 91, 95
<u>Coverage: 0;</u>

FULL COVERAGE

PARTIAL COVERAGE

CANDIDATE SUMMARY

Here are the main insights regarding exam stress tips:

- Students shared various methods for handling stress, including the use of **meditation apps** to promote relaxation and focus *[79,11,46,53,54]*.

- The **25-5 minute Pomodoro Technique** was discussed for its structure and possible positive impact on productivity and well-being *[79,80]*.

- A structured schedule for study and breaks was discussed as crucial for preventing stress and promoting effective exam preparation *[80,23]*.

Coverage Score	Citation Score	Joint Score
50	**51**	**22**

Figure 2: Example evaluation of a candidate summary (right) for its coverage of reference insights (left). Each reference insight is assigned a `Coverage Score` by mapping it to a single candidate bullet. A mapped bullet's citations are used to calculate the `Citation Score`. The total score is the average across reference insights. See Appendix A.7 for four additional examples.

peat across documents and citing source documents. The task resembles long-form question-answering (Fan et al., 2019) and query-focused summarization (Zhong et al., 2021; Vig et al., 2022).

In the following section, we describe the Haystack synthesis, the steps taken to ensure the quality of our benchmark, and the task framing.

3.2 Haystack Generation

Subtopic and Insight Generation (Figure 1, left) One of the main motivations behind synthetically generating documents is to precisely control information distribution in the documents.

A Haystack centers around a *topic* (e.g., "three students discuss strategies for an upcoming exam"). The first step generates a list of potential subtopics that can occur in documents about the topic. Subtopics are generic (e.g., students discussing study techniques, or how to manage stress). There are two subtopic requirements: (1) each subtopic should be *distinctive* and unique, such that no two subtopics overlap thematically, and (2) subtopics should be *expandable* into at least three distinct insights that are specific to the subtopic. Appendix A.1 goes over the quality assurance to ensure the satisfaction of these requirements.

In a second step, each subtopic gets instantiated into a list of specific *insights*, or facts that will be placed into the documents of the Haystack. Insights are defined as statements that contain specific information that may appear in a document about a given subtopic. Insights are expected to mention a number, a date, or an entity. For example, in the "Managing stress" subtopic, an insight can be: "a student explaining what the 25-5 Pomodoro technique is to the others". Crucially, insights should be specific, independent of each other, and solely relevant to a single subtopic. Appendix A.1.2 goes over the quality assurance to ensure insight quality.

The idea of breaking down documents into smaller information units has proven beneficial in recent work for both automatic and human evaluation (Min et al., 2023; Liu et al., 2022). Concretely, we use an LLM to generate subtopics and insights, optionally include context documents to provide seed ideas to the LLM, and aim to generate 10 subtopics, each with about 5-10 insights.

Document Generation (Figure 1, center)

The Haystack is synthesized one document at a time. For each document, we randomly select a set of insights across subtopics, and instruct an LLM to generate a document that must include all selected insights. The number of insights to include per document varies based on the domain, targeting 750 words of content per document (or roughly 1,000 tokens). By generating 100 documents, a Haystack totals on the order of 100k tokens. Appendix A.1 details quality assurance implemented during synthesis and two validation annotations evaluating the reliability of subtopic-to-insight and insight-to-document mappings.

Evaluation of the SummHay task relies on pre-

cise knowledge of the mapping between insights and documents in the Haystack. We implement over five domain-specific verification processes during the synthesis of the Haystack to ensure that the expected mapping is sound. Manual inspection and the high performance of human annotators on the final task, shown in Section 5.3 provide evidence of the quality of the resulting Haystacks.

3.3 Haystack Summarization (Figure 1, right)

Having generated Haystacks following the above protocol, we can now leverage them for the Summary of a Haystack task. Using an LLM, we transform each subtopic into a query (e.g., "What do the students discuss regarding stress management?").

Each system completing the task is instructed to generate a summary to answer the query (which focuses on a single subtopic), which must be in bullet-point format. Crucially, we instruct the LLM on the number of bullet points that the summary should contain, which matches the number of insights of the subtopic. Although this can appear as a simplifying assumption, this important design choice allows us to control for the length of generated summaries, which are a known confounding factor in summarization evaluation (Liu et al., 2022). We find in Section 5 that this choice effectively controls the length of generated summaries. The prompt also instructs the system to cite source documents in each of its bullet points, in a specific bracketed format (e.g., [1,2]), using document identifiers provided in the Haystack. The bullet-point structure and specific citation format are the foundation for our evaluation, detailed in Section 4.

3.4 SummHay Benchmark

We instantiate the above protocol across two domains: conversations and news, as these two domains are common test beds for summarization (Hermann et al., 2015; Gliwa et al., 2019). For each domain, we generate 5 Haystacks, and the average Haystack length is 93k tokens. Each Haystack consists of on average 9.20 subtopics, each averaging 6.75 insights, for a total of 62 insights per topic. For the news domain, we leverage the documents from Huang et al. (2023) as the seed context documents. Regarding LLM choice, we rely on a combination of GPT-3.5 and GPT-4o and specify additional details in Appendix A.1. Annotations described in Appendix A.1.2-A.1.3 confirm the accuracy of the subtopic-to-insight and document-to-insight mappings.

Method	Cov. Corr.	Link Acc.	Cost ($)
Manual Annot.	0.770	95.0	$325
Gemini-1.5-pro	**0.751**	**89.3**	$15.1
GPT-4o (9FS)	0.719	89.2	$26.1
GPT-4o	0.716	88.9	$6.9
Claude 3 Opus	0.677	87.9	$23.8
Claude 3 Haiku	0.498	87.7	$0.4
GPT3.5	0.495	86.7	$1.3

Table 1: Reproducibility and cost of manual and automated evaluation for SummHay. We compute coverage correlation, linking accuracy, and evaluation cost.

4 Evaluation Protocol

We first define task metrics (illustrated in Figure 2), establish the reproducibility of manual annotation, and then assess the quality of automated evaluation.

4.1 Evaluation Metrics

Coverage Metric Given a candidate subtopic summary, we extract each bullet point (split on line breaks) and want to measure the overlap between these bullets and the subtopic's reference insights. To do so, we iterate over each reference insight and assess whether the reference insight is *fully*, *partially*, or *not* covered in any of the candidate bullet points. For each insight, the summary receives a score of 100 for full coverage, 50 for partial coverage, and 0 otherwise. The final coverage score of a summary is the average coverage on all the insights of the subtopic, such that it ranges from 0 to 100. In Figure 2, the top reference insight is fully covered by the second candidate bullet, the second insight is partially covered by the first candidate bullet, and the third insight is not covered in the summary. The Coverage Score is: $(100 + 50 + 0)/3 = 50$.

While coverage can be framed as a binary decision (no coverage vs. full coverage), we believe that incorporating partial coverage allows for a more fine-grained analysis. Often a candidate summary bullet may partially include multiple insights, which motivated the inclusion of a partial coverage level. For example, a candidate summary insight might list several stress-management apps in a single bullet point. In the case where each app and its explanation corresponds to a separate insight in the reference summary, these insights are only partially covered. Applying a no vs full coverage metric would thus penalize a model that largely covers the topic.

Citation Metric Because documents of the Haystack are synthesized, each reference insight can be traced to a gold-standard set of documents

that contain the insight. When a summary's bullet covers a reference insight, we can compare generated citations to this reference set of cites. For each partially or fully covered reference insight, cited documents are extracted from the paired summary bullet point using a regular expression ([.*]), and we measure the precision and recall between the generated and gold-standard cites. The Citation Score of a reference insight is calculated as the F1 score of the precision and recall, giving equal weight to both. In short, a system must be both precise and thorough in its citing to achieve a high Citation Score. The Citation Score of an entire summary is then the average insight Citation Score of all reference insights that were covered by the system. In Figure 2, the average Citation F1 of the two covered bullets is: $(29 + 73)/2 = 51$.

Requiring that systems produce thorough citations (i.e., cite all documents relevant to an insight) might appear unnatural at first hand. Several writing domains that involve citation (such as scholarly writing) do not emphasize citation thoroughness, as it is often impractical: it is simply impossible to cite all papers that support a statement, as the universe of citable papers is large. Because Haystacks in our experiments contain 100 documents, this enables us to evaluate not only if systems can achieve high precision in their attribution, but also perform the task thoroughly, which is more challenging. We note that besides the added technical challenge, thoroughness in attribution can be beneficial to end users, for example helping them assess which fact is supported by a broader group of sources and which is more marginal (supported by 4 vs. 1 source).

Joint Metric The Joint Metric pieces Coverage and Citation Scores together, measuring whether a candidate summary both covers the expected insights and cites documents appropriately. The Joint Score of a summary is calculated by iterating over each reference insight and multiplying its coverage score and citation scores (assigning a Citation Score of 0 in case the insight is not covered). The Joint Score of a summary ranges from 0 to 100. In Figure 2, the summary's Joint Score is: $(100 * 0.29 + 50 * 0.73 + 0 * 0)/3 = 21.8$. Appendix A.7 provides four additional examples on the same subtopic with added details.

4.2 Annotation Reproducibility

To establish the reproducibility of the evaluation protocol, two authors of the paper and two profes-

sional annotators independently annotated a common subset of 35 summaries, annotating for the coverage of 240 insights within the summaries. The Manual Annotation row of Table 1 reports the inter-annotator agreement levels on the 240 coverage judgments. Coverage Score averaged a Pearson correlation of 0.77 across pairs of annotators, indicating a strong level of agreement between participants. When annotators agree that a reference insight is covered, they agree on which candidate bullet originates the coverage in 95% of cases (Linking Accuracy). In short, **annotators have strong agreement on which reference insights are covered by a given candidate summary and also agree on the specific bullets in the candidate summary that cover each insight.**

Annotating a single summary takes 4 minutes on average. To reduce evaluation costs and scale experiments, we next investigate LLM-based evaluation as an alternative to annotation.

4.3 Automatic Metric Validation

We recruited two professional annotators to annotate 200 candidate summaries (100 for each of the news and conversational domains) paired with reference insights. Annotators were compensated at $25/hour, and annotation required a total of 13 hours of work, for a total of $325.

We prepared a prompt that contains task instructions and an example summary which has a fully covered, a partially covered, and an uncovered insight (see Appendix A.2). We evaluated 5 LLMs as evaluators: GPT3.5, Claude 3 Haiku, Opus, GPT-4o, and Gemini-1.5-pro. In Table 1, we report evaluator performance in terms of correlation on the insight-level coverage scores, linking accuracy, which measures whether LLMs can attribute the coverage to the correct bullet point, and the cost of evaluating the 200 summaries. We find that two models, GPT-4o and Gemini-1.5-pro achieve a strong positive correlation (0.7+) with the human annotation. We select the GPT-4o model as our evaluator, as it achieves high evaluation correlation at a fraction of the cost of Gemini-1.5-pro, and does not have strict rate limits in place. We attempt one improvement by preparing a prompt with 9 few-shot examples (three of fully, partially, and uncovered insights each), and report the result as GPT-4o (9FS). Although the increased number of examples does lead to minor correlation improvements, it comes at a large cost increase, and thus we finalize the auto-evaluation setting as using the

Summarizer	Coverage Score (↑)							Citation Score (↑)							Joint Score (↑)							#W_b
	Rand	Vect	LongE	KWs	RR3	Orac	Full	Rand	Vect	LongE	KWs	RR3	Orac	Full	Rand	Vect	LongE	Kws	RR3	Orac	Full	
GPT3.5	36.2	45.8	46.0	48.4	51.9	56.2	–	9.3	15.2	15.0	15.9	16.8	23.0	–	3.6	7.3	7.2	7.9	9.0	13.2	–	28.2
Claude 3 Haiku	49.9	64.9	62.3	63.4	66.6	72.1	62.3	13.4	25.1	25.5	26.5	28.8	35.6	14.1	7.1	17.4	17.2	17.7	20.1	26.8	9.2	31.9
GPT4-turbo	49.4	61.0	56.7	61.2	61.8	67.1	57.9	17.9	28.6	28.1	31.1	31.8	41.4	5.5	9.6	18.7	16.9	20.1	20.6	28.9	3.2	37.9
Claude3.5 sonnet	57.0	69.3	70.8	71.3	75.7	80.0	71.8	18.4	30.5	30.7	30.7	32.5	39.9	18.3	11.5	22.2	23.0	23.1	25.8	32.7	13.8	27.7
Command-r	47.0	54.8	53.5	56.0	55.2	60.4	50.3	17.7	34.6	34.9	37.5	40.4	53.8	30.9	8.9	19.6	19.6	21.9	23.6	33.9	16.2	33.1
Gemini-1.5-flash	49.7	58.1	58.9	61.8	62.6	65.1	59.4	17.4	31.9	31.8	34.2	43.6	51.7	32.8	9.2	19.4	20.0	22.0	28.7	34.9	21.0	31.6
Command-r +	44.2	56.4	53.1	56.2	58.9	61.0	44.5	20.4	41.7	41.7	43.1	46.8	60.2	19.9	9.6	24.7	24.0	25.7	29.3	38.3	9.7	25.5
o1-mini	50.8	64.1	63.6	63.2	67.3	71.8	56.5	18.6	37.5	35.8	40.1	43.0	53.9	9.8	9.9	25.2	24.1	26.6	30.5	40.3	5.7	34.9
Mistral-large2	51.8	65.0	61.5	64.0	67.8	73.0	31.2	18.9	37.1	37.0	37.2	41.2	54.5	1.8	10.3	25.0	24.3	25.0	29.3	40.8	0.6	34.1
Claude 3 Sonnet	55.8	70.6	69.7	72.1	73.1	77.7	73.6	18.0	34.9	36.6	37.3	41.1	51.7	23.5	11.0	26.1	27.2	28.5	31.4	41.2	18.3	33.5
Claude 3 Opus	56.5	72.4	69.6	72.5	76.5	81.4	76.2	17.7	34.3	35.8	37.3	39.4	50.7	22.3	11.1	26.5	26.7	28.6	31.9	42.5	18.0	29.3
GPT-4o	54.0	67.1	67.8	66.6	70.4	76.6	66.1	21.9	38.4	38.0	38.6	41.3	54.6	16.2	12.6	27.3	27.6	27.3	30.8	43.4	11.4	36.5
Gemini-1.5-pro	53.0	63.5	64.9	63.6	68.4	67.6	70.0	21.9	43.1	44.5	46.6	49.7	64.1	51.0	12.3	28.6	31.0	30.8	36.0	44.6	37.8	30.5
o1-preview	56.8	73.9	72.1	74.2	78.7	83.9	73.7	24.9	51.5	51.3	52.5	57.0	67.4	35.3	15.1	40.3	39.1	41.1	47.3	58.2	27.4	34.6
Human Perf.	–	–	–	–	–	74.5	–	–	–	–	–	–	73.9	–	–	–	–	–	–	56.1	–	29.7

Table 2: Summary of a Haystack results of human performance, RAG systems, and Long-Context LLMs. Results are reported using three metrics: Coverage (left), Citation (center), and Joint (right) scores. Full corresponds to model performance when inputting the entire Haystack, whereas Rand, Vect, LongE, KWs, RR3, Orac correspond to retrieval components RAG systems. Models ranked by Oracle Joint Score. For each model, #W_b report the average number of words per bullet point.

original prompt and the GPT-4o evaluator.

In Appendix A.3, we assess whether automatic evaluation using GPT-4o could cause two types of biases: first, whether it could systematically favor or disfavor summaries generated by a family of models (such as GPTs), and second whether it could be partial to summaries of a certain length. We find no sign of systematic bias in the LLM-based evaluation in the case of our protocol.

5 Results

5.1 Experimental Settings

As illustrated in Figure 1 (right), we evaluate both long-context LLMs that directly access the full Haystack and RAG systems where a retriever filters the Haystack down to documents it perceives as relevant, which get passed to a generator (LLM). By default, documents in the Haystack are ordered in a single arbitrary order.

Full-Context Summarization We test a range of recent LLMs with context lengths longer than an individual Haystack, including Cohere's Command-R and Command-R+, Google's Gemini-1.5-pro and Gemini-1.5-flash (Reid et al., 2024), OpenAI's GPT4-turbo, GPT-4o, o1-preview and o1-mini, Anthropic's Claude models (haiku, sonnet, opus, and 3.5sonnet), and Mistral-large2. We also include GPT-3.5 exclusively in the RAG setting, as its context length is 16k tokens.

Retrieval-Augmented Summarization We evaluate RAG systems to reduce the Haystack input size. All retrieval models receive the query and all Haystack documents and must produce a query relevance score for each document. We sort the documents in reverse order according to the query relevance score and select the first 15k worth of document tokens. We chose 15k enables us to experiment with generators that have a 16k context window (GPT-3.5-turbo). We experiment with a total of six retrievers, each implemented as a separate query relevance score function. Under **KWs**, the document score is the number of overlapping keywords, extracted by NLTK, between the document and the subtopic query. We compare embedding methods that compute the cosine similarity between each document and the subtopic query, **Vect**, a SentenceTransformers (Reimers and Gurevych, 2019) embedder and **LongE** (Zhu et al., 2024), which extends standard embedders to cover longer input contexts, and include the Rerank3 (**RR3**) model from Cohere (Inc., 2024). We also include a **Rand** baseline that randomly assigns relevance scores, and an oracle setting ranker **Orac**, whose score is the number of subtopic insights that appear in a given document. The last two provide lower- and upper-bound retrieval quality estimates.

5.2 Benchmark Results

Table 2 summarizes the SummHay results across long-context, in the Full column, and RAG settings for all 17 Summarizers included in our study.

Coverage scores – which measure the presence of expected insights in a system's output summary – range from 36.2% when using a random retriever and GPT3.5-turbo as the summarizer, to 83.9% using the Oracle retriever with the o1-preview generator. The choice of retriever impacts the Cov-

erage score, with Random and Oracle retrievers leading to the worst and best scores, respectively, for almost all summarizers. Yet, top-performing LLMs like Claude3-opus achieve strong Coverage scores (70%+) under most retrieval settings, including Full context. In other words, **achieving strong coverage of key insights in a large corpus of text does not require retrieval, given a sufficiently capable long-context LLM**.

Citation scores – which account for both the precision and the thoroughness of the model's attribution back to source documents – present a complementary narrative. The lowest citation score often occurs in the full-context setting, with citation quality on par with random retrieval. On the other hand, as retrieval improves (for example from Random to RR3 to Oracle), citation scores increase. In a nutshell, **for use-cases where citation quality is important, optimizing retrieval is paramount**: it removes irrelevant documents from the summarizer's context, narrowing and focusing options for citation. Gemini-1.5-pro stands out as an outlier: it is the only model that achieves comparable Citation scores in RAG and long-context settings.

Taking Coverage and Citation into account, the Joint Score provides the complete system performance on SummHay. As expected, all Summarizers perform best with the Oracle retriever, an unrealistic setting intended to evaluate score ranges.

All models except for Gemini-1.5-pro achieve their realistic best performance in the RAG setting using the RR3 retriever. The higher relative performance of the more advanced RAG retriever RR3, developed for enterprise search and RAG settings, aligns with expectations compared to simpler retrievers. This result confirms the validity of our SummHay as a test-bed for holistic RAG evaluation; newer, more advanced RAGs can be benchmarked in an end-to-end fashion on SummHay, measuring direct impact on output quality.

We still observe two large gaps: all RAG systems underperform the Oracle setting, indicating ample room for improvements in RAG systems, and **models achieve very low Joint scores in the full-context setting (10-20), indicating that SummHay is an unsolved task for long-context LLMs.** Gemini-1.5-pro stands with strong ability to cite in the full-context setting, achieving the only realistic score above 35 on the benchmark.

OpenAI's o1-preview achieves the overall best realistic performance when combined with the RR3 retriever with a 47.3 joint score, largely ahead of the second best: Gemini-1.5-pro in long-context setting (37.8). o1-preview achieves this impressive performance by topping both coverage and citation scores, whereas other top-performing models tend to trade-off between the two metrics (i.e., Claude 3 Opus tends to perform strongly on coverage, whereas Gemini-1.5-pro tends to shine at citation). **This confirms the importance of the Joint Score, measuring system ability to concurrently retrieve relevant insights and attribute the insights to appropriate documents.**

The right-most column of Table 2 shows each system's average bullet point length (number of tokens). Several systems (Gemini-1.5-pro and Claude 3 Opus) average 30 words per bullet, close human-written bullet points (29.7). Others (GPT-4o, GPT4-turbo) are more verbose, at 36-38 words per bullet point. In Appendix A.3 we confirm that verbosity does not bias evaluation: succinct methods can achieve strong performance on SummHay.

Appendix A.5 breaks down Citation Scores into Precision and Recall. No system excels at either precision or recall but we do observe trade-offs. For example, Claude models generally achieve higher precision and lower recall, whereas Command-r + and GPT-4o favor recall over precision.

5.3 Estimating Human Performance

We estimate human performance on the task by recruiting two annotators to perform the task. We first define the setting in which annotation was conducted, then go over the results.

The participants performed the task in the Oracle setting, only viewing documents relevant to the query they are currently summarizing, as it reduces the volume of text they must read from about 100,000 tokens to 15,000 tokens. We assume this effectively reduces the amount of time required for annotation by a factor of 5-6, but this remains unverified, as it is impractical to conduct human annotation on the full Haystack.

In total, two annotators participated in writing a total of 10 summaries, five for subtopics in the conversational domain, and five for subtopics in the news domain. Although this represents a subset of the 92 subtopics in the entire SummHay benchmark, we believe it represents an unbiased estimate of human performance on the benchmark.

Appendix Figure 3 aggregates results across the ten annotation sessions. Overall, participants make steady progress during the sessions, with both Coverage and Citation rising rapidly in the first 90

| | **Document Order** | | | |
Summarizer	Top	Bottom	Random	Sensitivity
GPT-4o	13.8	**24.1**	11.4	12.6
Claude3 Opus	20.4	**28.0**	18.0	10.0
Gemini-1.5-pro	**47.1**	38.9	37.9	**9.2**

Table 3: Joint Scores of LLMs in the Full Context Setting, based on how documents are sorted. Documents can be in Random order or sorted such that relevant ones are at the Top or Bottom of the context window.

minutes, and then at a slower pace in the last 30.

The Citation Score corresponds to an F1 measure, and we also report on the Precision and Recalls of the Citations. We find that citation precision averages close to 80.0 throughout the session, whereas recall rises steadily during the session.

Table 2 includes the summarized final scores in contrast to system scores, showing that a human annotator can significantly outperform LLMs and RAG systems on the SummHay task, as the human joint score (56.1) is significantly higher than the best system performance (44.6). We caution the reader not to consider our estimate of human performance as an upper bound, as we believe that with more time and explicit instructions to double-check their work, annotators could further increase their scores. We solely intend the human performance to be a reference point for achievable performances on the benchmark, and we expect future systems to tie and surpass human performance on the task. Appendix A.4 provides further detail on guidelines, recruitment, and task framing.

5.4 Position Bias Sensitivity

In the Full Context experiment results (Table 2, `Full` columns), documents in the Haystack are ordered arbitrarily, with relevant documents in the top, middle, and bottom portions of the context window. Prior work (Huang et al., 2023; Chang et al., 2023; Chen et al., 2023b; Ravaut et al., 2023) has reported that models exhibit a `position bias` that leads them to put more importance on information in the context window's extremities. SummHay offers a framework to study position bias systematically. In Table 3, we report the results of the Position Bias experiment, in which we run the SummHay experiment with the top-3 performing models on sorted Haystacks, where relevant documents to a subtopic are either all at the Top or Bottom of the context window. Similar to prior work, we find that all three models exhibit position bias, with GPT-4o and Claude3 Opus performing better when relevant documents are at the bottom of the context window, and Gemini-1.5-pro favoring Top Haystacks. We compute a `Position Sensitivity` score as the maximum absolute difference in Joint Score between the Random ordering and Top and Bottom conditions. Future systems should strive to attain minimal sensitivity on SummHay, as document ordering is often arbitrary in real-world applications.

5.5 No-Document Baseline

Because insights and documents are generated synthetically, strong model performance could be tied to models being able to "hallucinate" insights without relying upon documents, and simply imagining what the insights could be. This could be particularly problematic for models used during Haystack synthesis, such as GPT-4o.

To assess this limitation, we instruct GPT-4o to imagine what the insights should be for any given subtopic, without providing any of the documents in the prompt, which we call the *no-document baseline*. In this baseline, GPT-4o achieved a coverage score of 23.2, less than three times what it achieves in conditions that provide documents, confirming that even when an LLM is involved in SummHay synthesis, it cannot perform the task accurately without having access to documents. The no-document baseline has a citation score of 0.05: despite lacking access to documents, the model still attempts to cite nonexistent sources when following prompt instructions.

6 Conclusion

In this work, we address the challenges of evaluating long-context LLMs and RAG systems by introducing the SummHay benchmark task, synthesized to assess the ability of systems to precisely summarize large sets of documents. The SummHay task requires generating summaries that accurately cover and cite insights relevant to a particular query. Our comprehensive evaluation reveals that current models struggle with this task; even in an oracle document setting, models lag behind human performance by more than 10 points. We believe that SummHay provides a robust framework for evaluating long-context systems and will encourage researchers to utilize SummHay to drive progress toward systems that can match or surpass human performance in long-context summarization.

7 Limitations

Task Upper Bound In Section 3.2 and Appendix A.1, we detail our data pipeline and efforts to ensure the quality of our dataset. This includes insight subtopic verification and verifying the inclusion of only specified insights for each document. Despite our efforts to prevent overlap among insights and guarantee the presence of insights in the documents generated, errors may be introduced from using LLMs in scaling this data synthesis, making achieving a perfect joint score of 100 likely unachievable. Although we estimate human performance in a simplified setting, we do not determine the task upper bound. However, we show significant room for improvement between the realistic full-context and RAG settings and the Oracle setting.

Simplifying Assumptions in Data Synthesis The assumptions made when generating Haystack documents likely introduce artificial signals that simplify the task. For example, in order to maximize control over the data synthesis process, Haystack documents are generated independently; no dependencies or cross-references exist among the documents. However, in a real-world multi-document summarization task, documents may link or refer to each other, and there may be temporal between documents. We believe the introduction of more realistic assumptions can further increase the difficulty of the task, and we hope that future work will take our synthesis processes as a starting point for such improvements.

Controlling for Verbosity When generating summaries, we specify the desired number of insights for the LLM to generate. Furthermore, we do not control for or penalize the verbosity of the summaries, and summaries with longer insights may result in higher coverage. Not specifying the number of summary insights needed per query will result in a more difficult task, and we leave a study of the potential trade-offs between verbosity, human preference, and overall scores for future work.

Reliance on Automated Evaluation Although we do not observe significant bias towards a particular family of models, as shown in Appendix A.3, the results in Table 1 demonstrate that there is room for improvement both in coverage and linking evaluation. Gemini-1.5-pro, in addition to being more costly than GPT-4o, had a rate limit which inhibited its use in our study. Although highly-cost effective, non-LLM based NLI and relevance

metrics (Chen and Eger, 2023; Liu et al., 2023) were not tested; Chen et al. (2023b) found worse performance among smaller NLI models for the related task of unrelated sentence identification on long-form question answering.

Model Choice The generation models included in our study are all closed-source models. Although these closed-source models currently generally outperform open-sourced models, this performance comparison can be task-dependent (Chen et al., 2023a). We exclude high-performing open-sourced models such as Llama-3 (AI@Meta, 2024) as the original models cannot handle the minimal 16k context window necessary for our RAG experiments. Restricting the length of the retrieved documents to, for example, 8k would remove too many of the insights for a given subtopic; by allowing up to 15k tokens in the RAG setting, we find that the oracle citation F1 achievable with this context length is 0.84 averaged across insights, which we believe strikes a balance between the reduction of input size and the feasibility of the task. We leave an analysis of RAG systems across retrieved input lengths, as well as models specifically designed for output citation (Menick et al., 2022) for future work and encourage and benchmarking of longer-context, open-sourced models on SummHay.

Ethical Considerations

The models and datasets utilized in the project primarily reflect the culture of the English-speaking populace. Gender, age, race, and other socio-economic biases may exist in the data, and models trained on these datasets may propagate these biases. Text generation tasks such as summarization have previously been shown to contain these biases.

In Section 4 and Section 5, we recruited professional annotators to perform evaluation, or directly attempt the task. We ensured to remunerate the participants fairly ($25/hour). Participants could communicate with us to voice concerns, work at their own pace, and choose to stop working on the project at any time. Finally, we ensured to anonymize the annotations (annotator identity is instead marked as annotator1, annotator2, etc.).

In our work, we relied on several datasets as well as pre-trained language models. We explicitly verified that all datasets and models are publicly released for research purposes and that we have proper permission to reuse and modify the datasets.

References

AI@Meta. 2024. Llama 3 model card.

Chenxin An, Shansan Gong, Ming Zhong, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. L-eval: Instituting standardized evaluation for long context language models. *arXiv preprint arXiv:2307.11088*.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Manik Bhandari, Pranav Narayan Gour, Atabak Ashfaq, Pengfei Liu, and Graham Neubig. 2020. Re-evaluating evaluation in text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yapei Chang, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2023. Booookscore: A systematic exploration of book-length summarization in the era of llms. *arXiv preprint arXiv:2310.00785*.

Hailin Chen, Fangkai Jiao, Xingxuan Li, Chengwei Qin, Mathieu Ravaut, Ruochen Zhao, Caiming Xiong, and Shafiq Joty. 2023a. Chatgpt's one-year anniversary: Are open-source large language models catching up? *arXiv preprint arXiv:2311.16989*.

Hung-Ting Chen, Fangyuan Xu, Shane A Arora, and Eunsol Choi. 2023b. Understanding retrieval augmentation for long-form question answering. *arXiv preprint arXiv:2310.12150*.

Yanran Chen and Steffen Eger. 2023. Menli: Robust evaluation metrics from natural language inference. *Transactions of the Association for Computational Linguistics*, 11:804–825.

Elizabeth Clark, Shruti Rijhwani, Sebastian Gehrmann, Joshua Maynez, Roee Aharoni, Vitaly Nikolaev, Thibault Sellam, Aditya Siddhant, Dipanjan Das, and Ankur P Parikh. 2023. Seahorse: A multilingual, multifaceted dataset for summarization evaluation. *arXiv preprint arXiv:2305.13194*.

Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. *arXiv preprint arXiv:2309.13345*.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021a. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021b. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Mingqi Gao and Xiaojun Wan. 2022. DialSummEval: Revisiting summarization evaluation for dialogues. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5693–5709, Seattle, United States. Association for Computational Linguistics.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

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

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701.

Nan Hu, Jiaoyan Chen, Yike Wu, Guilin Qi, Sheng Bi, Tongtong Wu, and Jeff Z Pan. 2024. Benchmarking large language models in complex question answering attribution using knowledge graphs. *arXiv preprint arXiv:2401.14640*.

Kung-Hsiang Huang, Philippe Laban, Alexander R Fabbri, Prafulla Kumar Choubey, Shafiq Joty, Caiming Xiong, and Chien-Sheng Wu. 2023. Embrace divergence for richer insights: A multi-document summarization benchmark and a case study on summarizing diverse information from news articles. *arXiv preprint arXiv:2309.09369*.

Cohere Inc. 2024. Introducing rerank 3: The next generation of search relevance. Accessed: 2024-06-10.

Ehsan Kamalloo, Aref Jafari, Xinyu Zhang, Nandan Thakur, and Jimmy Lin. 2023. Hagrid: A human-llm collaborative dataset for generative information-seeking with attribution. *arXiv preprint arXiv:2307.16883*.

Gregory Kamradt. 2023. Needleinahaystack.

Yekyung Kim, Yapei Chang, Marzena Karpinska, Aparna Garimella, Varun Manjunatha, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2024. Fables: Evaluating faithfulness and content selection in book-length summarization. *arXiv preprint arXiv:2404.01261*.

Kalpesh Krishna, Erin Bransom, Bailey Kuehl, Mohit Iyyer, Pradeep Dasigi, Arman Cohan, and Kyle Lo. 2023. Longeval: Guidelines for human evaluation of faithfulness in long-form summarization. *arXiv preprint arXiv:2301.13298*.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. 2024. In search of needles in a 10m haystack: Recurrent memory finds what llms miss. *arXiv preprint arXiv:2402.10790*.

Wai-Chung Kwan, Xingshan Zeng, Yufei Wang, Yusen Sun, Liangyou Li, Lifeng Shang, Qun Liu, and Kam-Fai Wong. 2023. M4le: A multi-ability multi-range multi-task multi-domain long-context evaluation benchmark for large language models. *arXiv preprint arXiv:2310.19240*.

Philippe Laban, Andrew Hsi, John Canny, and Marti A Hearst. 2020. The summary loop: Learning to write abstractive summaries without examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5135–5150.

Philippe Laban, Tobias Schnabel, Paul N Bennett, and Marti A Hearst. 2022a. Summac: Re-visiting nli-based models for inconsistency detection in summarization. *Transactions of the Association for Computational Linguistics*, 10:163–177.

Philippe Laban, Chien-Sheng Wu, Lidiya Murakhovs'ka, Xiang Chen, and Caiming Xiong. 2022b. Discord questions: A computational approach to diversity analysis in news coverage. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5180–5194.

LangChain. 2024. Multi-needle in a haystack.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

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

Dongfang Li, Zetian Sun, Xinshuo Hu, Zhenyu Liu, Ziyang Chen, Baotian Hu, Aiguo Wu, and Min Zhang. 2023a. A survey of large language models attribution. *arXiv preprint arXiv:2311.03731*.

Xinze Li, Yixin Cao, Liangming Pan, Yubo Ma, and Aixin Sun. 2023b. Towards verifiable generation: A benchmark for knowledge-aware language model attribution. *arXiv preprint arXiv:2310.05634*.

Yifei Li, Xiang Yue, Zeyi Liao, and Huan Sun. 2024. Attributionbench: How hard is automatic attribution evaluation? *arXiv preprint arXiv:2402.15089*.

Yixin Liu, Alexander R Fabbri, Pengfei Liu, Yilun Zhao, Linyong Nan, Ruilin Han, Simeng Han, Shafiq Joty, Chien-Sheng Wu, Caiming Xiong, et al. 2022. Revisiting the gold standard: Grounding summarization evaluation with robust human evaluation. *arXiv preprint arXiv:2212.07981*.

Yixin Liu, Alexander R Fabbri, Yilun Zhao, Pengfei Liu, Shafiq Joty, Chien-Sheng Wu, Caiming Xiong, and Dragomir Radev. 2023. Towards interpretable and efficient automatic reference-based summarization evaluation. *arXiv preprint arXiv:2303.03608*.

Daniel Machlab and Rick Battle. 2024. Llm in-context recall is prompt dependent. *arXiv preprint arXiv:2404.08865*.

Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, Francis Song, Martin Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingham, Geoffrey Irving, et al. 2022. Teaching language models to support answers with verified quotes. *arXiv preprint arXiv:2203.11147*.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation. *arXiv preprint arXiv:2305.14251*.

Xuanfan Ni, Hengyi Cai, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, and Piji Li. 2024. Xl bench: A benchmark for extremely long context understanding with long-range dependencies. *arXiv preprint arXiv:2404.05446*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Mathieu Ravaut, Shafiq Joty, Aixin Sun, and Nancy F Chen. 2023. On context utilization in summarization with large language models. *arXiv e-prints*, pages arXiv–2310.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

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*.

Dingjie Song, Shunian Chen, Guiming Hardy Chen, Fei Yu, Xiang Wan, and Benyou Wang. 2024. Milebench: Benchmarking mllms in long context. *arXiv preprint arXiv:2404.18532*.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Liyan Tang, Philippe Laban, and Greg Durrett. 2024. Minicheck: Efficient fact-checking of llms on grounding documents. *arXiv preprint arXiv:2404.10774*.

Jesse Vig, Alexander Fabbri, Wojciech Kryscinski, Chien-Sheng Wu, and Wenhao Liu. 2022. Exploring neural models for query-focused summarization. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1455–1468, Seattle, United States. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yunshu Wu, Hayate Iso, Pouya Pezeshkpour, Nikita Bhutani, and Estevam Hruschka. 2023. Less is more for long document summary evaluation by llms. *arXiv preprint arXiv:2309.07382*.

Xiang Yue, Boshi Wang, Ziru Chen, Kai Zhang, Yu Su, and Huan Sun. 2023. Automatic evaluation of attribution by large language models. *arXiv preprint arXiv:2305.06311*.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. 2023. Alignscore: Evaluating factual consistency with a unified alignment function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11328–11348.

Huajian Zhang, Yumo Xu, and Laura Perez-Beltrachini. 2024a. Fine-grained natural language inference based faithfulness evaluation for diverse summarisation tasks. *arXiv preprint arXiv:2402.17630*.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, et al. 2024b. Infinitybench: Extending long context

evaluation beyond 100k tokens. *arXiv preprint arXiv:2402.13718*.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021. QMSum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921, Online. Association for Computational Linguistics.

Dawei Zhu, Liang Wang, Nan Yang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2024. Longembed: Extending embedding models for long context retrieval. *arXiv preprint arXiv:2404.12096*.

A Appendix

A.1 Haystack Synthesis Details

Below we specify additional details regarding our data synthesis pipelines. For the news domain, we leverage GPT-4o for all data synthesis steps. We found it necessary to leverage this high-performing LLM due to the longer seed context documents that subtopics and insights are generated from. For the conversation domain, we leverage GPT-4o to generate subtopics and insights, and for any LLM-based verification step. Conversation generation (conditioned on selected insights) is completed using GPT-3.5-turbo.

Furthermore, we employ verification steps to ensure that subtopics and insights are distinct and precisely mapped to Haystack documents. When generating documents given a set of insights, we do not want other insights to "*leak*" into the document, as that would reduce the quality of the Haystack and task. Below we list the verification steps taken across the conversation and news domain. Differences in domain characteristics and the seed used to generate Haystacks necessitate per-domain verification steps.

A.1.1 Subtopic Verification

In the news domain, to ensure distinct insights and subtopics we first prompt an LLM to identify any overlapping or duplicate subtopics and remove these subtopics. This helps ensure that when querying relevant insights for a subtopic, insights can only belong to one of the distinct subtopics generated.

In the conversation domain, we use manual inspection to verify the distinctness of the subtopics and regenerate subtopic candidates (at temperature

$T = 1$) until obtaining a list where each subtopic feels qualitatively unique.

A.1.2 Insight Verification

For the news domain, we prompt the LLM to remove duplicate insights. After producing an initial set of insights for the subtopics, we take all insights and prompt an LLM to categorize each insight into one of the subtopics. As insights are initially generated for a particular subtopic, at this step we ensure that no insight can fit into another subtopic. We thus remove any insights for which the categorized subtopic differs from the one it was initially generated for.

In the conversation domain, we iterate over subtopics sequentially and use a prompt to generate the list of insights for one subtopic. In the prompt, we provide not only the target subtopic but also all other subtopics and insights, instructing the LLM to avoid such subtopics and insights, and only propose insights that are distinct and unique in contrast to those. Manual inspection from the authors reveals that: as long as the subtopics are confirmed to be unique, and that the insights are enforced to be specific (and include entities), very little overlap occurs across subtopic insights.

To assess the validity of the subtopic-to-insight mapping, we hired an annotator unfamiliar with the SummHay generation process and tasked them to assign each insight to one or several subtopics that are most relevant. We ran the above experiment by recruiting a professional annotator (non-author of the paper) and having them annotate all insights of 4 topics (2 in the conversation domain, 2 in the news domain, randomly selected from our 10 total topics). Since there are roughly 50-80 insights per topic, this corresponds to a total of 245 annotations.

The results of the annotation are summarized in Table 4. Accuracy is slightly higher for news: 92.4%, and slightly lower for conversation: 89.5%.

The findings indicate that the annotator assigned less than 1% of insights to multiple subtopics. The annotator achieved an overall accuracy of 91% in assigning the insight to its expected subtopic. (The number is slightly higher for news: 92.4%, and slightly lower for conversation: 89.5%). In general, this indicates that although the Subtopic to Insight synthesis process is not perfect, it is largely one-to-one. During the iterations of the project, we found that one way to enforce that insights more uniquely map to a subtopic is by ensuring that insights are long, specific, and rich in entities. This helps in-

	Conversation	News	Both
Insights Annotated	114	131	245
% Insights Assigned to 2+ Subtopics	0.9	0.8	0.8
% Insight → Subtopic Accuracy	89.47	92.37	91.02

Table 4: Quality verification of Subtopic-to-Insight mapping across conversation and news domains.

crease "the space of possible insights" and reduce the chance of collision and ambiguity.

Ambiguous cases do occur. For example, the insight: "The sales rep mentions a recent benchmark study that shows Salesforce's performance metrics compared to top competitors, highlighting Salesforce's speed and data reliability." was mapped by the annotator to the subtopic: "customer asking how Salesforce compares with competitors" whereas the synthesis process expected the insight to belong to: "sales rep sharing success stories and case studies". Some ambiguity is bound to exist in any such dataset, but we believe that a rate of less than 10% is acceptable and doesn't severely impact task validity.

A.1.3 Document Verification

In the news domain, to ensure a precise mapping of insights to documents in which an insight is present, we prompt an LLM to label whether any of the insights, across subtopics, other than those sampled for that document are found in the document. If any are found, we regenerate the document, asking the LLM to remove the sentence(s) containing the extraneous insights. This procedure is repeated until no extraneous insights are found, up until 5 iterations. For a similar purpose, we prompt an LLM to label whether the insights sampled for a given document are indeed found in the document. If not, we regenerate the document to add sentences that contain this insight, up until 5 iterations.

To understand the relative quality of the mapping between insights and documents, we conducted an annotation. The task consisted of reading a document and selecting from a list of insights which ones they believe are present in the document, and which ones are not.

We hired a professional annotator to read through 10 documents (one from each of the ten Haystacks: 5 conversation, 5 news), and annotate for each which insights they believe occur in the document. Results are summarized in Table 5. We note that slightly lower is achieved in the in the conversation domain vs the news domain.

Overall, the annotator achieved perfect precision:

Title	Conversation	News	Both
Num Samples	70	85	155
Accuracy	94.29	98.82	96.77
Precision	100	100	100
Recall	80.00	97.14	90.91
F1	88.89	98.55	95.24

Table 5: Quality verification of Document-to-Insight mapping across conversation and news domains.

when they believe an insight is in the document, it always is. The annotator achieves an average recall of 90 (slightly lower in the Conversation domain, and higher in News), meaning that in some rare cases, the annotator believes an insight is not present, even if it is expected to be present according to our synthesis process. The overall accuracy of 96% confirms the reliability of our automatic mapping process.

. We find that after 5 iterations of editing, discrepancies in insights by the LLM were primarily paraphrasing or partial detail upon manual inspection.

In the conversational domain, we generate the documents iteratively, one *chapter* at a time, where each chapter is intended to introduce a single insight. When expanding an insight into a chapter, we generate a candidate chapter and use GPT-4o to classify whether the candidate chapter indeed covers the expected subtopic and insight. If not, we regenerate the candidate chapter up to ten times, and otherwise, we accept the candidate chapter into the document and proceed with the next chapter. We find that in practice, the generation process requires 5 iterations less than 1% of the time to generate a chapter that GPT-4o can correctly assign to the expected insight.

A.2 Evaluation Prompt

Below, we list the prompt we use for automatic evaluation in the SummHay task, as described in Section 4.

```
You are given a list of bullet points (
    each with a unique number), and a
    specific reference insight. Your
    objective is to determine whether
    the reference insight is covered in
    any of the bullet points. You must
    further determine if the insight is
    partially covered ("PARTIAL_COVERAGE
    ") or fully covered ("FULL_COVERAGE
    ") by the bullet points. If the
```

	Evaluator Model		
	GPT-4o	Opus	Gem-1.5-Pro
Summarizer Model Bias			
Claude 3 Sonnet	0.027	-0.001	-0.012
Gemini-1.5-flash	0.051	0.024	-0.009
GPT3.5	0.009	0.050	0.048
Claude 3 Opus	0.059	0.034	0.043
Gemini-1.5-pro	0.088	0.065	0.065
GPT4-turbo	0.075	0.091	0.056
Command-r +	0.064	0.128	0.071
Claude 3 Haiku	0.092	0.108	0.071
GPT-4o	0.097	0.102	0.106
Average	0.062	0.067	0.049
Summary Length Bias			
Length to Score Corr.	-0.122	-0.174	-0.178
Length to Delta Corr.	0.02	-0.051	-0.081

Table 6: Results of analysis of potential bias in automated evaluation. We explore potential bias caused by what model is used, which is reported in differences of scores with human annotation, and bias due to the length of the summary, which is reported as a correlation. An unbiased evaluator model should achieve a bias close to zero on both analyses.

```
insight is not covered at all, you
    must return "NO_COVERAGE". See
    examples below:

[[FEW_SHOT_EXAMPLES]]

Now complete the task for the following
    insight and bullet points:

Reference Insight:
[[INSIGHT]]

Bullet Points:
[[BULLETS]]

Requirements:
- Do not hallucinate that the insight is
     covered by the bullet points if it
    is not.
- Your response should only be the JSON
    output in the format above, such
    that it can directly parsed by
    Python's json module. DO NOT OUTPUT
    ANY EXPLANATION OR ANYTHING THAT IS
    NOT THE JSON RESPONSE.
```

A.3 Automatic Results Bias

There is a concern that since we propose to use an LLM to automate the evaluation of the SummHay experiment, the choice of the evaluator model might affect the validity of the results if such a model has a systematic bias in its judgment. We evaluate the possible presence of two biases. First, whether the automatic evaluation could favor outputs of one model family over the other (e.g., GPT-4o systematically favoring outputs of the GPT* family).

To study this, we perform an automatic evaluation of score bias by calculating the difference (Δ) between the Coverage Score according to the auto-evaluation, and the Coverage Score from the manual annotation. A positive Δ indicates that the auto-evaluation assigned a higher score than the manual evaluation, and a negative Δ indicates the opposite. We calculate the average Δ for each Summarizer model in our experiment, and inspect the bias of three evaluator models: GPT-4o, Claude3 Opus, and Gemini-1.5-pro.

The top portion of Table 6 summarizes the Summarizer model bias analysis results. First, we find that auto-evaluation results almost always have a positive bias, indicating that on average, the auto-evaluation overestimates Coverage by roughly 5 points across models. Second, we find that evaluator models tend to have a positive bias for top-performing Summarizer models (e.g., GPT-4o, Claude3 Opus, and Gemini-1.5-pro), but do not systematically prefer outputs from a specific model family. In fact, Claude3 Opus seems to be particularly critical of Claude3 model outputs (with biases very close to zero). All models have a strong positive bias towards GPT-4o outputs, but it does not translate to bias for a model family. **Overall, we find no evidence of systematic bias of automatic evaluation that would favor one model family over the other.** The analysis does reveal a pattern of overestimating coverage by an average of 5 points, which should be taken into account when interpreting the results.

In a second analysis, we study whether automatic evaluation leads to favoring summaries based on their length.

To study length bias, we first see whether a summary's score correlates with its length, as measured by the number of words divided by the number of bullet points. We divide by the number of bullet points as each query requires a different number of bullet points, which directly affects length of the summary in a way that is not controlled by the LLM. By measuring length bias based on the length of individual bullet points, we remove this confounding variable.

In the `Length to Score Corr.` row of Table 6, we find that there is a slight negative correlation (-0.12 to -0.178) between a summary's score and the number of words in its bullet points. This correlation could be explained by results from Section 5, which showed that several of the top-performing models (Claude3-opus, Gemini-1.5-pro) generate the shortest bullet points. In other words, short bullet points could achieve higher scores not because they are short, but because they were generated by better models. To remove this confounding variable, we measure whether automatic score Δ (computed above) correlates with bullet point length. This analysis, summarized in te `Length to Delta Corr.` row of Table 6, indicates a non-existent correlation between bullet-point length, and whether the automatic evaluator was biased in its scoring (-0.081 to 0.02). In conclusion, we do not find evidence that using automatic evaluation with our evaluation protocol will cause length bias in our results, which would systems to generate shorter or longer bullets points.

A.4 Details on Establishing SummHay Human Performance

To establish task duration, we considered an average reading speed of 200 words per minute. Carefully reading the documents (roughly 12,000 words) would therefore require one hour. Accounting for the need to write the summary, and scan multiple times over documents to identify and cite insights, annotators were told they had a maximum of two hours to complete the task. Participants could take breaks (i.e., pause their work) and finish the task early if they felt the task was completed. After their initial sessions, participants were asked whether two hours seemed appropriate to complete the task without rushing, and both agreed. Participants sometimes used the entirety of the two hours, and in other cases completed the task in as little as 80 minutes.

We relied on professional annotators known and trusted by our research group (based on performance on previous annotation work), and they were compensated at 25 USD per hour. One of the annotators participated in the annotation of automated summaries (and therefore had access to reference
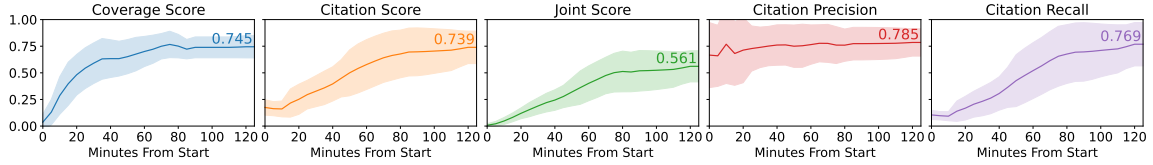
Figure 3: Estimates of human performance on the SummHay task, plotted over time as participants complete the task in the Oracle setting during two-hour sessions.

insights for certain subtopics), and we ensured that this annotator only performed the summarization task for subtopics and document sets they had not seen during that annotation, ensuring they had no prior knowledge of the documents in the Haystack.

Participants were given all documents in a shuffled order and were instructed to read them carefully and summarize any insight that seemed to be repeating across documents. Participants were told the number of present insights (similar to LLM prompt in our experiment). In practice, we found that participants chose to write slightly more bullet points than the number they were given.

Regarding citation, participants were instructed to be as thorough as possible and to explicitly look for additional citations when they had identified an insight.

Regarding tool use, participants were allowed to use string search (i.e., `Ctrl+F`), but were prohibited from copying the text from the documents, and were explicitly instructed they should not use ChatGPT or equivalent LLM-based interfaces in any way to assist them with the task. Because this cannot be strictly enforced practically, we rely on trusted professional annotators to complete the task.

Participants were instructed to gradually write their summary in a text box in the annotation interface, and we recorded the progress on the summary during each annotation summary. We then performed auto-evaluation using the same settings used in our benchmarking experiments on the final summary of the session, as well as a summary every 10 minutes during the session.

A.5 Citation Precision & Recall Analysis

The Citation Score in the SummHay benchmark is an F1 calculation between the set of cites generated by a system in a given bullet point, and the expected cites of the matched reference insight, based on knowledge from the Haystack generation of what documents include the insight. F1 is chosen as a measure to ensure that systems balance between

Summarizer	Precision		Recall		F1 (Citation)	
	Orac	Full	Orac	Full	Orac	Full
GPT3.5	46.7	–	17.9	–	23.0	–
Claude 3 Haiku	49.3	24.7	31.6	24.2	35.6	14.1
GPT4-turbo	62.3	14.1	35.7	3.8	41.4	5.5
Claude 3 Opus	66.0	30.7	45.8	24.0	50.7	22.3
Gemini-1.5-flash	57.8	38.2	54.5	44.2	51.7	32.8
Claude 3 Sonnet	67.3	42.2	47.2	19.9	51.7	23.5
Command-r	58.9	38.9	55.9	31.7	53.8	30.9
GPT-4o	65.7	28.0	51.0	13.0	54.6	16.2
Command-r +	67.6	24.2	60.8	21.2	60.2	19.9
Gemini-1.5-pro	76.2	59.0	60.9	52.4	64.1	51.0
Human Perf.	78.8	–	82.4	–	76.7	–

Table 7: Breakdown of Citation Precision and Recall of models on the SummHay Benchmark, combined into an F1 Score (Citation Score). Numbers are reported for the Full Context and the Oracle settings of SummHay.

precise and thorough cites.

Table 7 reports the precision and recall of all systems on the benchmark, as well as the F1 (i.e., the Citation score) to shed light on how different systems balance between precision and recall.

A.6 Model Access Details

For each model in our study, we specify its model card and how it was accessed.

We access the Google models Gemini-1.5-pro (`gemini-1.5-pro-preview-0514`) and Gemini-1.5-flash (`gemini-1.5-flash-preview-0514`) through Vertex AI [2].

We include three OpenAI models in our study: GPT-3.5-turbo (`gpt-3.5-turbo-0125`), GPT-4-turbo (`gpt-4-turbo-2024-04-09`), and GPT-4o (`gpt-4o`). All models were accessed through OpenAI's official API[3].

Cohere summarizers Command-R (`cohere.command-r-v1:0`) and Command-R+ (`cohere.command-r-plus-v1:0`) were accessed through Amazon Bedrock[4], while Rerank3 (`rerank-english-v3.0`) was accessed

---
[2] https://cloud.google.com/vertex-ai
[3] https://github.com/openai/opeai-python
[4] https://aws.amazon.com/bedrock/

through Cohere's official API[5].

Anthropic models were also accessed through Amazon Bedrock: Claude 3 Haiku (`anthropic.claude-3-haiku-20240307-v1:0`), Claude 3 Sonnet (`anthropic.claude-3-sonnet-20240229-v1:0`), and Claude 3 Opus (`anthropic.claude-3-opus-20240229-v1:0`).

The embedders Vect (`sentence-transformers/all-mpnet-base-v2"`) and LongEmbed (`dwzhu/e5-base-4k`) were accessed through SentenceTransformers (Reimers and Gurevych, 2019) and huggingface's transformers library (Wolf et al., 2019), respectively.

A.7 Additional Output Examples

Figure 4 provides four examples of real summary outputs from different RAG pipelines on a common subtopic related to managing stress when preparing to an exam. For each summary, we also report on the Coverage, Citation and Joint scores, as calculated by the LLM-based automatic evaluation. We add color coding and bolding to facilitate the interpretation of the evaluation.

A.8 Additional Discussion

We point to several additional areas for future work.

English-centric   While our data pipeline can be extended to non-English languages with access to a seed scenario in a given language, our benchmark was developed only on English and may be more reliable in English. However, the task is language-agnostic, and future work can create a multilingual version of the SummHay task, similar to efforts such as Seahorse (Clark et al., 2023).

Beyond Relevance   We restrict our data synthesis process and analysis to target summarization relevance, but we believe that similar data procedures and evaluations could be applied to factual consistency. We leave an extension of our pipeline and analysis of model outputs along other dimensions such as coherence, efficiency (brevity), or factuality to future work.

Focus on Factoid-Style Insights.   Our Haystack synthesis process incentivizes the creation of specific insights that focus on a number or entity. Specificity helps simplify evaluation, and ensure we can achieve reproducible automatic evaluation. Yet real-world scenarios might have less clear-cut

insights, with different documents only partially overlapping on insights, or with potentially disagreeing conclusions (e.g., some people like the Pomodoro Technique while others don't). Prior work has shown that NLP methods struggle in such cases of coverage diversity (Laban et al., 2022b; Huang et al., 2023), and including such discord within the Haystack could yield more complex and realistic tasks.

Investigating Low Scores.   Manual inspection reveals different failure modes in low-scoring summaries, including (1) retrieving insights that are not relevant to the query (other subtopics), (2) framing very high-level information as a specific insight (e.g., "all the participants are polite to each other"), (3) hallucinating insights not directly supported in the documents. We do not systematically evaluate the frequency of each failure, but future work can explore this more systematically, for example using efficient NLI-based alignment (Laban et al., 2022a; Zha et al., 2023; Tang et al., 2024).

---

[5] https://docs.cohere.com/reference/rerank

REFERENCE INSIGHTS

• One student shares that they do 10 minutes of **deep breathing exercises** each night before going to bed. Reference docs: 8, 32, 46, 53, 69, 91, 95

• A student mentions that taking a **30-minute walk** each afternoon helps clear their mind and manage stress levels. Reference docs: 11, 20, 48, 54, 80, 91

• One student suggests taking a 5-min break after every 25 min of studying, and that the **Pomodoro technique** is helpful. Reference docs: 8, 32, 37, 48, 79, 83, 95

• A student recommends using the **meditation app 'Calm'** that they use for 15 minutes each morning to manage stress. In docs: 11, 30, 46, 53, 79, 80

• Another student shares they write down **3 positive affirmations** about their capabilities each day. Reference docs: 13, 27, 37, 54, 68, 83

Retriever: Oracle - Generator: GPT-4o

Coverage: 70 - Citation: 64 - Joint: 46

- They discussed various relaxation techniques to manage stress, such as deep breathing exercises, which were mentioned as beneficial for reducing anxiety and improving focus **[8]**[11]**[46][53][91]**.
- Walking and physical activity were commonly recommended as stress-relief strategies, with multiple participants highlighting the benefits of taking short walks to clear their minds and reduce stress levels **[11]**[53]**[54][80][91]**.
- The Pomodoro technique, where one studies for a set time (typically 25 minutes) followed by a short break, was suggested as a useful method to maintain concentration and prevent burnout **[8]**[11]**[37]**[53][54]**[79][83][95]**.
- Mindfulness and meditation practices, particularly using apps like 'Calm,' were shared as effective ways to manage stress before and during exam preparation **[11][53][79]**[83][91].
- The importance of taking regular breaks and incorporating positive affirmations into daily routines to boost morale and reduce stress was emphasized, as these small adjustments can make a significant difference in maintaining a positive mindset **[37]**[46]**[83][13]**.

Retriever: Rerank3 - Generator: Claude3 Opus

Coverage: 90 - Citation: 33 - Joint: 32

- The students shared various relaxation techniques to manage exam stress, such as deep breathing exercises before sleeping[54][83], and meditation using the 'Calm' app for 15 minutes each morning[54]**[79]**[91].
- Some students mentioned using the Pomodoro Technique, studying for 25 minutes followed by a 5-minute break.[27][30]**[79]**
- Incorporating physical activity into routines, such as taking a 30-minute walk each afternoon, was mentioned as a way to clear the mind, and reduce stress levels.**[54][91][11]**
- Positive affirmations and mindfulness practices were suggested to help students stay calm, confident, and focused.**[37][54]** Writing down three positive affirmations about their capabilities each morning was one specific technique shared.**[37]**

Retriever: Random - Generator: Gem-1.5-Pro

Coverage: 30 - Citation: 43 - Joint: 13

* Deep breathing exercises can help clear the mind and reduce anxiety. [4]**[46][95]**
* Meditation apps like 'Calm' can promote relaxation and focus. **[46][53][80]**
* Short walks outside or stretching provide a refreshing break from studying. [46][74]**[80]**
* Mindfulness practices can set a positive tone for the day and help manage stress. [53]
* Engaging in enjoyable activities like reading, watching movies, or spending time with friends and roommates offers a healthy distraction from exam preparation. [53][95]

Retriever: Vector - Generator: GPT3.5

Coverage: 30 - Citation: 21 - Joint: 6

- Tips for managing exam stress were shared among the students during the study group sessions [54][80][46].
- Strategies such as practicing mindfulness, deep breathing exercises, and taking short breaks were suggested to help reduce anxiety and stay focused [54][80]**[46]**.
- The importance of finding balance between studying and self-care, such as incorporating physical activity and mindfulness, was highlighted to improve overall well-being during exam preparation **[54][80]**[46]
- The idea of implementing structured study breaks using the Pomodoro Technique to enhance productivity and prevent burnout was discussed **[54]**[80][46].
- Students discussed the significance of supporting each other, sharing tips, and creating a positive, collaborative study environment to alleviate exam stress and stay motivated [54][80][46].

Figure 4: Examples of five insights within a subtopic, and four SummHay outputs from RAG systems, including the final Coverage, Citation, and Joint scores, as calculated by our LLM-based automatic evaluation. We add color coding and bolding to facilitate the interpretation of the evaluation.