Sequential-NIAH: A Needle-In-A-Haystack Benchmark for Extracting Sequential *Needles* from Long Contexts

Yifei Yu, Qian-Wen Zhang†, Lingfeng Qiao, Di Yin, Fang Li, Jie Wang, Zengxi Chen, Suncong Zheng, Xiaolong Liang, Xing Sun
Tencent Youtu Lab, China
cowenzhang@tencent.com, felixyfyu@tencent.com

Abstract

Evaluating the ability of large language models (LLMs) to process lengthy contexts is critical, especially for retrieving query-relevant information embedded within them. We introduce Sequential-NIAH, a benchmark specifically designed to evaluate the capability of LLMs to extract sequential information items (known as *needles*) from long contexts. The benchmark includes three needle generation pipelines: synthetic-temporal, real-temporal, and real-logical orders, with context lengths ranging from 8K to 128K, which comprises 14,000 samples (2,000 for testing). To facilitate the evaluation of this benchmark, we trained an evaluation model that assesses the correctness of LLM responses by comparing their completeness and sequential consistency against the ground truth, which provides a more reliable evaluation metric than GPT-4 or Claude. We conducted experiments on six well-known LLMs, revealing that even the best-performing model achieved a maximum accuracy of only 63.50% on test set of this benchmark. Further analysis highlights the growing challenges posed by increasing the context length or the number of needles, underscoring substantial room for improvement of LLMs. Additionally, noise analysis validates the reliability and challenge of the benchmark, making Sequential-NIAH an important reference for advancing research on long text information extraction capabilities of LLMs.

1 Introduction

Enhancing LLMs' long-context understanding has been a key focus in Natural Language Processing (NLP). Recent models like Gemini-1.5 (Georgiev et al., 2024), GPT-4 (Achiam et al., 2024), Claude-3.5 (Anthropic, 2024), Qwen-2.5 (Team, 2024), GLM-4 (Zeng et al., 2024), Kimi (Moonshot), and DeepSeek-V2 (Liu et al., 2024a) have extended context lengths to millions of tokens while maintaining reasoning and comprehension capabilities. Meanwhile, several benchmarks have been exposed for long context understanding, including ∞Bench (Zhang et al., 2024b), L-Eval (An et al., 2023), LongBench (Bai et al., 2024), LongEval (Krishna et al., 2023), LooGLE (Li et al., 2024a), ZeroSCROLLS (Shaham et al., 2023) and FactGuard (Zhang et al., 2025). However, these benchmarks typically focus more on the model's global comprehension of long texts or the retrieval of specific information at certain locations. In reality, challenging problems often involve retrieving and integrating detailed information from multiple parts of a document to produce the optimal answer, which can generally be defined as *Needle-in-a-Haystack* (NIAH) tasks (gkamradt, 2023; Hsieh et al., 2024; Song et al., 2024; Li et al., 2024b).

Although existing NIAH benchmarks provide challenging needle-retrieval tasks within long contexts, they still fail to account for the sequential characteristic of *needles*—such as temporal or logical order. This oversight is particularly significant in real-world scenarios, where explicit demands exist, such as:

- *List all events involving suspect Tom in March 2024 in temporal order, based on a legal document.*
- *List all Microsoft equity transactions in temporal order, based on a financial report.*
- *Outline the detailed steps to obtain a senior building engineer certification in sequential order, as per a guideline.*

Figure 1 provides a simple example illustrating the sequential relationships among needles, simulating a more realistic and challenging NIAH task. For LLMs, it is essential to not only retrieve query-relevant items but also comprehend their sequential relationships and present them in the correct order.

†Corresponding Author.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 29439–29457
November 4-9, 2025

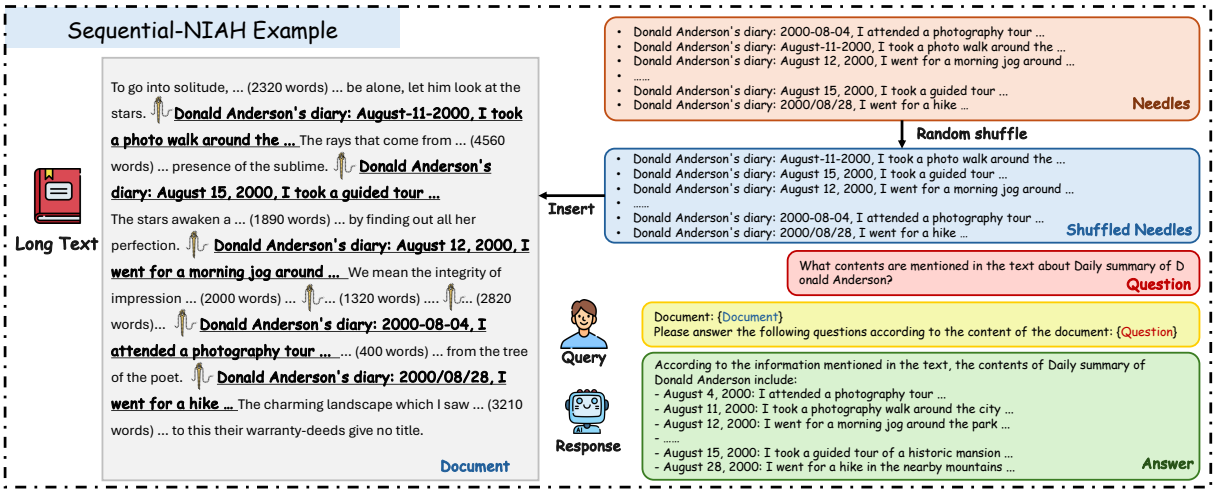


Figure 1: Sequential-NIAH example of a long text with shuffled needles with temporal order.

To supplement existing long context information retrieval evaluation methods, We introduce the Sequential-NIAH benchmark, which shuffles *needles* with temporal or logical order and inserts them into long contexts of varying lengths. Considering the completeness of the benchmark, we propose three different types of *needles* generation pipelines, including synthetic-temporal order, real-temporal order and real-logical order. Synthetic-temporal order *needles* are generated by fake entities, timestamps, and events. Real-temporal order *needles* are generated from the Temporal Knowledge Graph (TKG(García-Durán et al., 2018; Jin et al., 2020; liuhuanyong, 2022)), which can be used to build sequential temporal items based on the relationship of two entities overtime. Real-logical order *needles* are generated from a private open-domain QA resource. The first two types are mainly aimed at retrieving temporal sequence items, while the last type is mainly aimed at retrieving logical sequence items.

On the other hand, accurately evaluating the correctness of enumerated answer items is challenging to achieve. The common practice relies on assessment by powerful LLMs, such as GPT-4o, which increases evaluation costs and hinders researchers from conducting efficient benchmarking. Thus, we employed synthetic data to train an evaluation model. The synthetic dataset encompasses as diverse a range of incorrect answer types as possible (e.g., missing items, redundancies, errors, disordered items, etc.). These will be paired with reference answers as training data to teach the evaluation model to accurately identify incorrect responses. The validation results indicate that the evaluation model achieved an accuracy of 99.49% (much higher than GPT-4o and Claude) on the synthetic test set, which is a reliable evaluation tool for our benchmark.

Powered by our evaluation model, we assessed the accuracy of several popular LLMs on the benchmark. Results show the task is highly challenging, with the best model reaching only 63.50% accuracy. Longer contexts and more needles further increase difficulty of the task. A noise robustness analysis also confirmed the benchmark's reliability and challenge. Our key contributions are:

- **Sequential-NIAH benchmark**: A benchmark for evaluating LLMs' ability to retrieve sequential information from long contexts. It comprises three types of *needles* generation pipelines, covering both temporal-ordered and logical-ordered *needles* retrieval tasks, to simulate real-world application scenarios.

- **Evaluation model**: A model trained on synthetic data to facilitate the evaluation of LLMs' performance on our benchmark. An accuracy rate of 99.49% demonstrates the model's high reliability as an evaluation tool, which is more accurate, efficient and cheaper than GPT-4o and Claude.

- **LLMs' Limitations on Sequential-NIAH**: Experimental results indicate that all current LLMs have significant room for improvement on this task, struggling with the complexity of sequential information retrieval within long contexts.

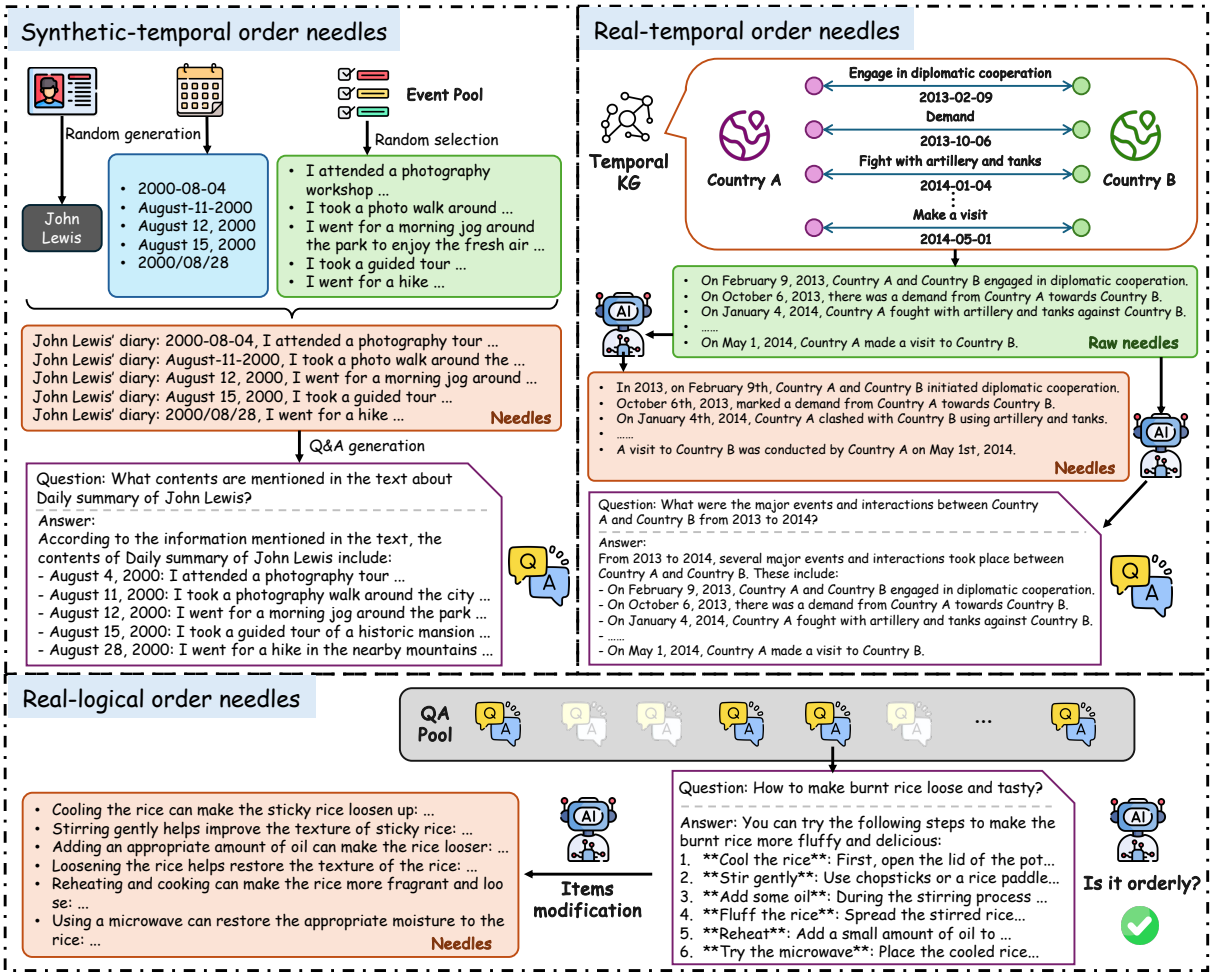


Figure 2: Three pipelines for sequential needles construction. Synthetic temporal order needles are generated by fake subjects, time stamps, and events (upper left). Real temporal orders are generated from the TKG (upper right). Real logical order needles are generated from a private open-domain QA resource (lower).

2 Related work

2.1 Long Context Language Models

Many techniques have been used to improve the context length that LLMs can handle. For instance, certain novel position embedding methods, such as ALiBi (Press et al., 2022), Position Interpolation (Chen et al., 2023), RoPE (Su et al., 2024) and its variants (Xiong et al., 2023; Liu et al., 2024b; Peng et al., 2023b). And some research aims to reduce context length by memory replay back-propagation (Wu et al., 2022), recurrent memory augmentation (Bulatov et al., 2024), and activation beacon (Zhang et al., 2024a). In addition, there are several methods to extend the context length by modifying the model architecture, such as Mamba (Gu and Dao, 2024), FLASHBUTTERFLY (Fu et al., 2023a), and RWKV (Peng et al., 2023a). Specifically, Gemini-1.5 supports a context length of 1 million tokens, and Kimi supports a context length of 2 million words. Most leading LLMs support a context length of at least 128k tokens, such as GPT-4o, Claude-3.5, Qwen-2.5, Qwen-3, and LLaMA-3.3. In this work, we will evaluate these LLMs with long-context analysis capabilities to assess their performance on our benchmark.

2.2 NIAH Benchmarks and Tasks

NIAH essentially represents a category of long-text information retrieval tasks, primarily assessing the capability of LLMs to retrieve multiple pieces of query-relevant information from long texts. The RULER (Hsieh et al., 2024) and Counting Starts (Song et al., 2024) benchmarks are designed with retrieval tasks at the word or character level (passwords or ★), where the problems involved are relatively clear and singular. NeedleBench (Li et al., 2024b) takes this one step further by designing more complex information on logical reasoning, such as descriptions of relationships between enti-

QA source					Long context source				
Pipeline	English	Chinese	Total	Proportion	Length	English	Chinese	Total	Proportion
Synthetic-temporal	3,000	3,000	6,000	42.86%	8k-16k	1,000	750	1,750	12.50%
Real-temporal	3,003	2,011	5,014	35.81%	16k-32k	2,000	1,750	3,750	26.79%
Real-logical	1,497	1,489	2,986	21.33%	32k-64k	2,000	1,750	3,750	26.79%
Total	7,500	6,500	14,000		64k-128k	2,500	2,250	4,750	33.92%
Proportion	46.43%	53.57%			Total	7,500	6,500	14,000	

Table 1: Information of QA source from three sequential needles synthetic pipelines (left) and long context source extracted from LongData-Corpus (right).

ties or kinship relationships, and inserting them into a long context. In Sequential-NIAH, we designed a challenging long-context information extraction task by needles with sequential characteristics, to better reflect real-world use cases.

2.3 Evaluation Model

The evaluation of natural language generation (NLG) is a vital but challenging problem in artificial intelligence (Gao et al., 2024). Its primary methods include the following four types: LLM derived metrics (Fu et al., 2023b; Jia et al., 2023), prompt-based LLMs (Ouyang et al., 2022; Luo et al., 2023; Gao et al., 2023; Liusie et al., 2024), fine-tuning LLMs (Xu et al., 2023; Ke et al., 2024; Zhu et al., 2023), and collaborative human-LLM evaluation (Zhang et al., 2021; Li et al., 2023). Commonly, simply designing prompts often fails to achieve optimal accuracy in evaluation, and fine-tuning on open-source models can enhance the accuracy of the evaluation model effectively. If manpower is sufficient, the combination of human effort with LLMs can further improve the reliability of the evaluation. To facilitate a reliable automatic evaluation of Sequential-NIAH tasks, we trained an evaluation model based on Qwen2.5-Instruct-32B, which provides a convenient and reliable evaluation tool for this benchmark.

3 The Sequential-NIAH Benchmark

The goal of Sequential-NIAH is to retrieve sequential needles from long contexts. Therefore, both the needles and the long contexts need to be prepared in advance. See the link[†] for details of the dataset.

3.1 QA source

We propose three sequential needles generation pipelines, as shown in Figure 2, which are used to build the question with sequential answer items, including synthetic-temporal order needles, real-temporal order needles and real-logical order needles. All are ultimately presented in the form of a question with multiple answer items (*needles*) inserted into a long text with length from 8K to 128K. Table 1 provides detailed information of the number and proportion of QA pairs constructed by different pipelines, collectively referred to as **QA source**. We adjusted the proportion of Chinese and English QA pairs to maintain each around 50%.

3.1.1 Synthetic-temporal order needles

Synthetic-temporal order needles refers to the synthesis of question-answer pairs using specific generation templates by combining subjects, event times, and event content. The question is usually posed about events that occur within a certain time period for a predefined fake subject, and the answer items are the synthetic events related to the fake subject listed in temporal order. In theory, this method can generate an unlimited number of qualified chronological question-answer pairs. We ensure the complexity of the task by designing various question templates and needle templates. The number of needles corresponding to a question can be set manually, and we randomly select the number of needles from 3 to 15.

3.1.2 Real-temporal order needles

From open source TKG datasets (ICEWS and FEG), we can extract real relationships between two different real entities change over time, which can be used to generate real-temporal order needles. In our pipeline, the relationships are rewritten (by GPT-4o) into question-answer pairs with temporal order answer items. The question is usually posed about the changes in relationships between two specific entities over a period of time. The amount of data that can be generated by this pipeline is limited by the size of the Graph, and the number of needles corresponding to a question depends on the number of relationships between two entities (from

[†] https://github.com/miraclefish/Sequential-NIAH-Benchmark.git

Algorithm 1: Data construction pipeline

Data: long text C, question Q, answer A, needles $N = [n_1, n_2, \ldots, n_k]$.
Result: $Query$ and $Answer$.

$[C_1, C_2, \ldots, C_{k+1}] \leftarrow$ Segment$(C, k+1)$;
$N \leftarrow$ Shuffle(N);
Long text with needles: $\hat{C} \leftarrow C_1$;
for $i \leftarrow 1$ **to** k **do**
  $\hat{C} \leftarrow \hat{C} + n_i + C_{i+1}$;
end
$Query \leftarrow$ Prompt$(\hat{C}, Q)$;
$Answer \leftarrow A$;
return $Query$ and $Answer$;

3 to 10 in this pipeline).

3.1.3 Real-logical order needles

In addition to items in temporal order, there are also cases where answers follow a precise logical order. To incorporate these into the benchmark, we filtered out question-answer pairs that meet the requirements from a private open-domain QA database with the help of GPT-4o. A total of 2,986 QA pairs are collected, whose answer items strictly adhere to a logical order. We manually conducted a sampling check on the filtered data to ensure the reliability of the QA filtering. Considering that directly inserting the answer items into the long context might seem abrupt (making it difficult to establish a direct connection between the question and the needles), we also use GPT-4o to rewrite the answer items to generate more naturally phrased needles. This ensures that when needles are inserted into the long context, they can still make connections to the question, maintaining the rationality of the task.

3.2 Long Context Source

To enhance the authenticity of the task, we use LongData-Corpus (yuyijiong, 2023), a real long text corpus, to construct the **long context source**. The corpus contains more than 100k pieces of Chinese and English long texts with lengths exceeding 8k characters, with the longest text exceeding 256k characters. The text content covers a wide range of materials such as academic papers, novels, legal documents, news, patents, government work reports, etc. This provides ample long context data for the construction of the benchmark. When preparing the long context source, to keep the language and quantity of long texts consistent with the QA source, we randomly sample long texts within different length ranges for each language (Chinese and English), as shown in Table 1. This forms a long context source that covers a wide enough range of topics, has a reasonable distribution of article lengths, and can match each QA pair in the QA source one-to-one.

	English	Chinese	Total
Train	5,400	4,600	10,000
Development	1,015	985	2,000
Test	1085	915	2,000
Total	7,500	6,500	14,000

Table 2: Dataset information of Sequential-NIAH Benchmark

3.3 Sequential-NIAH Sample Constructing

For a given long text and a QA pair, a specific Sequential-NIAH sample is constructed by randomly shuffling the order of needles in the answer of QA and inserting them into random positions within the long text. Subsequently, the sample is formatted into $Query$ (inserted long text and question) and $Answer$ (reference answer) forms using a designed prompt template, as shown in Figure 1. The detailed procedure is described in Algorithm 1. Ultimately, the dataset is partitioned into three subsets: training set, development set, and test set. Each subset contains data from diverse languages, varying text lengths, and distinct needles synthetic pipelines, embodying a Sequential-NIAH benchmark, as shown in Table 2.

4 Evaluation Model

Due to the complexity of benchmark evaluation, we hope to automate the evaluation of this task by training an evaluation model f_θ. For each question Q_i, the ground truth answer A_i and an corresponding answer B_i to be evaluated are provided to constitute an input $X_i = T_{eval}(Q_i, A_i, B_i)$, where $T_{eval}(\cdot)$ is a prompt template for answer evaluation. For each

	English	Chinese	Total
Train	3,000	3,000	6,000
Test	984	967	1,960
Total	3,984	3,967	7,960

Table 3: Dataset information for evaluation model training and test.

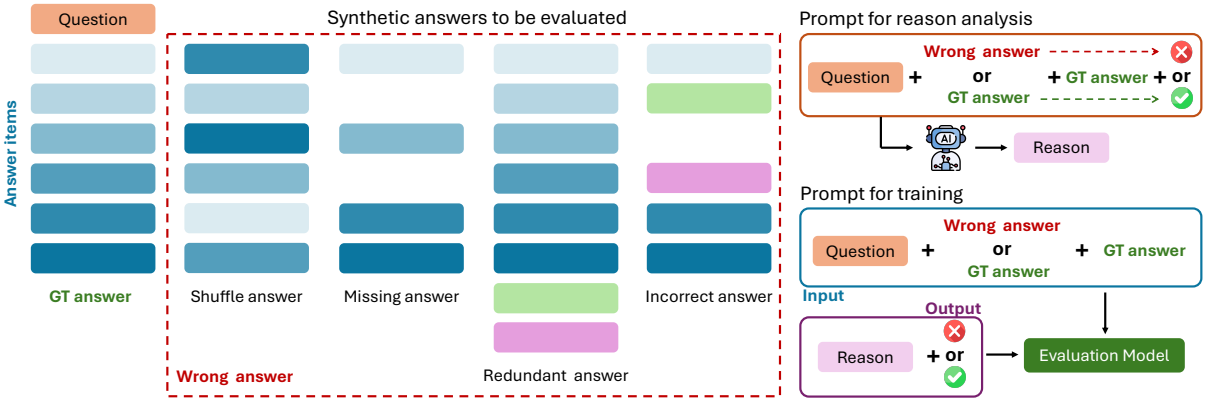


Figure 3: Synthetic different wrong answers for generating diversified training data for evaluation model training.

Needle types	Answer groups	No.	Claude-3.5 (%)	GPT-4o (%)	Ours (%)
Temporal order	GT	241	85.06	90.87	**95.85**
	Missing	235	93.62	97.45	**100**
	Redundant	265	77.74	85.28	**100**
	Incorrect	229	95.20	97.38	**100**
Logical order	GT	229	86.03	99.13	**100**
	Missing	249	93.17	**100**	**100**
	Redundant	266	85.34	99.25	**100**
	Incorrect	246	82.11	**100**	**100**
	Total/Avg.	1960	87.09	96.07	**99.49**

Table 4: Evaluation model performance on synthetic test data with various needle types and answer groups.

X_i, the label $Y_i = T_{res}(y_i, R_i)$ is constructed by the result ($y_i \in \{wrong, correct\}$) and the reason R_i, where $T_{res}(\cdot)$ is the prompt template for result analysis. To train the evaluation model, our objective is to learn the probability distribution of Y_i conditioned on X_i, i.e., $P(Y_i|X_i;\theta)$. And the loss function can be defined as:

$$\mathcal{L}(\theta) = \arg\max_{\theta} \sum_{i=1}^{N} \log P(Y_i|X_i;\theta) \quad (1)$$

To obtain the training data, four types of potential wrong answers are synthesized by changing GT answer items, including shuffled answer items (shuffled GT answer items), missing answer items (GT answer items with random missing items), redundant answer items (GT answer items and random redundant items), and incorrect answer items (random missing items and random redundant items coexist). For the last three groups of answers, we uniformly consider them as wrong answers. For the first group of answer with only shuffled items, if the question does not require the answer items to be output in specific order, it will be treated as correct answers; otherwise, it will be treated as an wrong answers.

Figure 3 provides an example of how to generate training data for evaluation model training. At first, four kinds of wrong answer are generated according to the ground truth (GT) answer. For each question Q_i we can get an answer pair by combining the GT answer A_i and the answer B_i to be evaluated. Then, Q_i, A_i, B_i and the known result y_i are organized into the reason analysis prompt template to get the reason by GPT-4o. Finally, the obtained reason R_i and y_i can be combined as the output Y_i, and Q_i, A_i and B_i will be organized as the input X_i for evaluation model training.

As shown in Table 3, a total of 6,000 samples are constructed to train the evaluation model, and 1,960 samples are used to evaluate its performance. The data used to train and test the evaluation model are randomly sampled from the QA source. We used Qwen2.5-Instruct-32B as the foundation for our evaluation model and performed full-parameter SFT training on it. We utilized the AdamW optimizer, setting the learning rate to 8×10^{-6} with 4 epoch. We set the warm-up ratio to 0.1 and the weight decay to 0.1.

5 Experiments & Results

5.1 Evaluation Model Performance

To demonstrate the need to train the evaluation model, we compared the performance of Claude-3.5, GPT-4o, and our evaluation model on 1,960 test samples using the same prompt templates (T_{eval} and T_{res}). The experimental results are shown in Table 4, and the results are divided into two groups (needles with temporal and logical order) for analysis. More detailed grouped results are presented in Table 7.

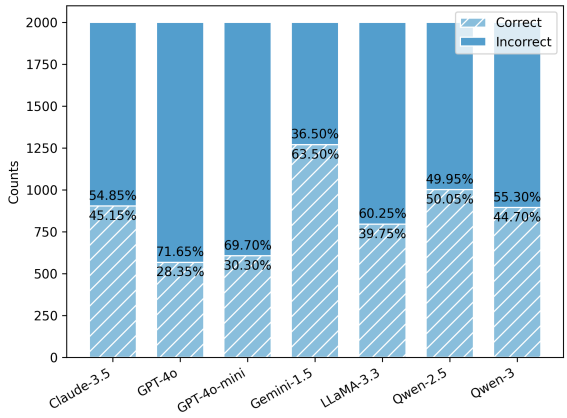


Figure 4: Results comparison of LLMs on all test data of Sequential-NIAH benchmark.

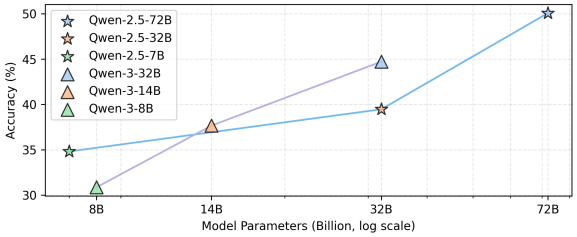


Figure 5: Performance of model size scaling on Qwen-2.5 and Qwen-3 series.

According to analysis, the evaluation model we trained achieved a total accuracy rate of **99.49%** in two groups of test data, which is much higher than GPT-4o and Claude-3.5. Our evaluation model achieved an accuracy of 100% in 7 answer groups. And the only 0.51% misjudgment of our evaluation model came from the model's slight confusion about whether the question requires listing answer items in temporal order (only occurring in shuffled GT answer items). This sufficiently demonstrates that using the model for automated evaluation of the benchmark is entirely feasible.

5.2 Benchmark Results of LLMs

To evaluate the performance of different LLMs on this benchmark, we conducted inference on 2,000 test samples using four closed-source models, including **Claude-3.5** (Claude-3.5-sonnet-20241022), **GPT-4o** (GPT-4o-20240806), **GPT-4o-mini**, **Gemini-1.5** (Gemini-1.5-pro), and three open-source models, including **Qwen-2.5** (Qwen2.5-72B-Instruct), **Qwen-3** (Qwen3-32B), and **LLaMA-3.3** (LLaMA-3.3-70B-Instruct).

Overview of results: Figure 4 illustrates the overall performance of LLMs on this benchmark. Gemini-1.5 exhibits the best performance, achieving an accuracy of 63.50%. Qwen-2.5 follows closely behind with an accuracy of 50.05%, while LLaMA-3.3, Qwen-3 and Claude-3.5 demonstrate comparable levels of performance. In contrast, GPT-4o-mini and GPT-4o perform poorly on this task. Moreover, experiments on Qwen-2.5 and Qwen-3 with size scaling show that performance degrades as the model size decreases, as shown in Figure 5.

Results across length groups: Figure a shows that the accuracy of most LLMs decreases as the text length increases. However, Gemini-1.5 and Qwen-2.5 maintain better and more stable performance. LLaMA-3.3 and GPT-4o decrease significantly with longer text.

Results across the number of needles groups: Intuitively, more needles will significantly increase the difficulty of this task. Figure b shows that all LLMs effects deteriorate as the number of needles increases. Surprisingly, Gemini-1.5 still maintains more stable accuracy compared with others.

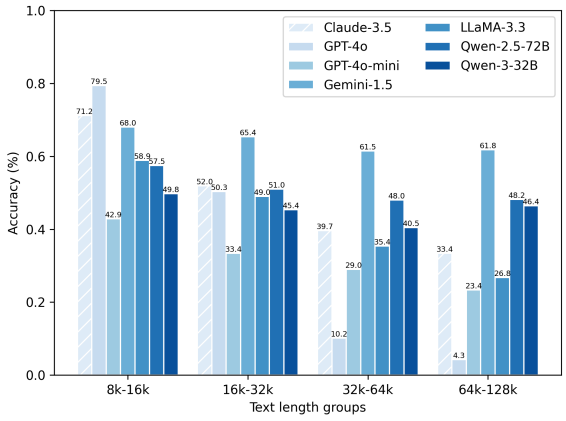
Results across needles generation pipeline groups: Figure c presents that most LLMs perform better on test data composed of real-logical order needles. It may be attributed to the fact that the questions in this group include some general knowledge, allowing LLMs to provide answers close to the GT based on their inherent capabilities, rather than retrieving from long texts. It also indicates that retrieving and listing information in temporal order from long texts is more challenging.

Results across language groups: Figure d depicts that the performance of the same model on test samples in different languages is generally consistent, indicating that language is not a key factor affecting the difficulty of the task.

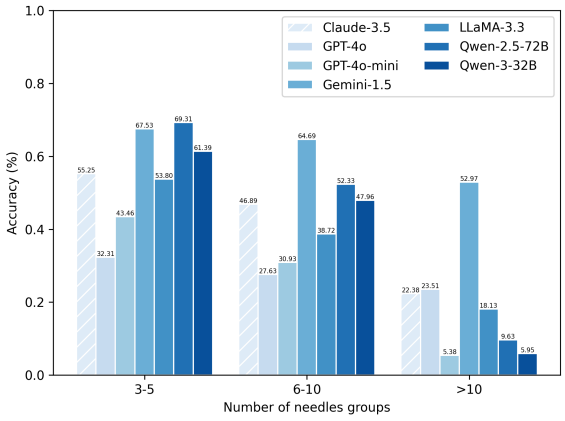
5.3 Noise Analysis of the Benchmark

In our investigation of this benchmark's characteristics, we selected 200 samples from the test set to conduct a noise analysis. Noise analysis, in this context, refers to evaluating the stability of various LLMs' performance when the needles or the long context face variations that may affect response. Specifically, we introduced four distinct types of noise to each sample:

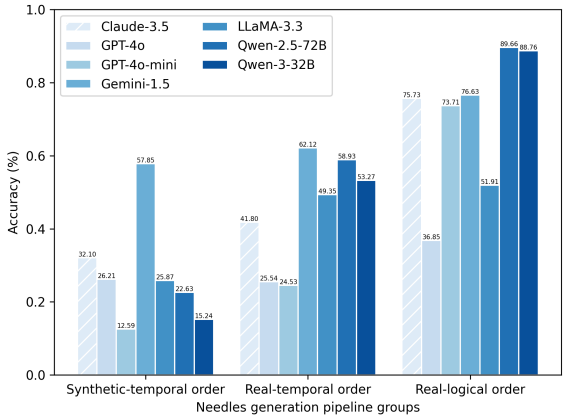
- **Tiny movement (TM)**: Each needle within the long text undergoes a slight positional shift, either forward or backward, by no more than two sentence positions. The order of



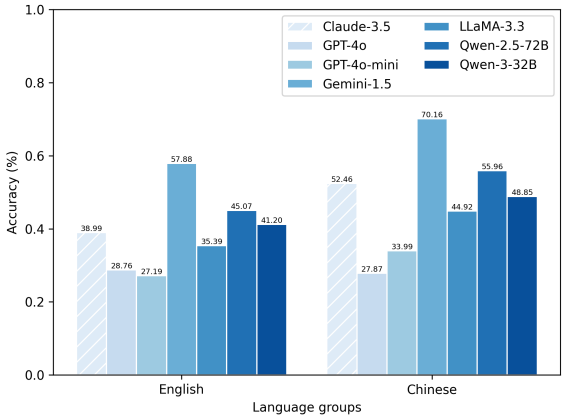
(a) Results across length groups.



(b) Results across the number of needles groups.



(c) Results across needles generation pipeline groups.



(d) Results across language groups.

Figure 6: Benchmark results of well-known LLMs on test data across different groups.

the needles within the long text remains unchanged

- **Significant movement (SM)**: Multiple needles within the long text are repositioned significantly, with their original sequence maintained. This simulates larger displacements, enabling analysis of the model's robustness to more pronounced positional alterations.

- **Reorder (RO)**: The positions of multiple needles remain static, but their sequence of appearance within the text is shuffled.

- **Semantic noise (SN)**: One or two synthetic needles that are semantically similar to one needle—yet cannot serve as a valid answer item—are inserted at somewhere in the long context, simulating scenarios that may confuse the model's judgment.

For each type of noise, we generated three variations of the original data, culminating in a total of 2400 noisy samples used for inference and evaluation. In this section, three specific models, **Gemini-1.5**, **Qwen-2.5**, and **LLaMA-3.3**, are subjected to the noise analysis experiments to discern their resilience and performance under these controlled perturbations.

Two metrics are employed for the noise analysis experiment: average accuracy (**Acc.**) and result

Model	Metrics	Ref. (%)	TM (%)	SM (%)	RO (%)	SN (%)	All (%)
Gemini-1.5	Acc.	62.50	65.00 ±2.55	63.12 ±1.89	64.12 ±1.80	53.88 ±7.36	61.31 ±6.58
	Cons.	-	56.50	52.00	47.50	41.50	17.50
Qwen-2.5	Acc.	51.50	48.50 ±2.00	48.25 ±2.25	48.00 ±2.68	43.00 ±6.10	45.88 ±3.73
	Cons.	-	67.50	65.50	64.50	64.50	42.00
LLaMA-3.3	Acc.	38.00	38.00 ±0.71	36.50 ±2.38	42.00 ±2.92	29.88 ±5.78	36.27 ±6.07
	Cons.	-	63.50	55.50	55.50	62.00	29.50

Table 5: Average accuracy and result consistency of LLMs with different noise assigned on test data. 'All' indicates that all noise groups are collectively included in the metric calculation.

consistency (**Cons.**). **Acc.** represents the mean accuracy across the original 200 test samples and additional test samples with introduced noise. **Cons.** assesses the stability of the model's responses by comparing the consistency of answers' evaluation result of the original 200 test samples with those of the noise-altered sets.

Average accuracy analysis: In Table 5, the reference accuracy (**Ref.**) represents the original accuracy achieved by the 200 samples drawn from the test set. Under full-noise conditions (column 'All'), the Acc. of the three models deviates from the reference values by 1.19% for Gemini-1.5, 5.62% for Qwen-2.5 and 1.73% for LLaMA-3.3. While exhibiting model-specific variance, all LLMs demonstrate quantitatively acceptable performance deviations (no more than 6%). It demonstrates that the test set exhibits consistent reliability in evaluating LLMs' ability on this benchmark, with evaluation results remaining robust against both minor and major needles variations.

Result Consistent analysis: Table 5 also shows that the LLMs exhibit varying degrees of response stability under noise influence, with Qwen performing best, LLaMA second, and Gemini worst. This may occur because LLaMA's responses contain a relatively high proportion of incorrect answers, and the introduction of noise fails to correct these errors, resulting in consistently higher error rates in its outputs. On the other hand, the more noise groups are introduced, the worse the Cons. becomes, which clearly demonstrates that noise can significantly impact the correctness of LLM responses, further highlighting the challenging of this benchmark.

6 Conclusion

We introduce Sequential-NIAH, a benchmark for evaluating LLMs on sequential information extraction from long texts (up to 128K tokens). It includes synthetic-temporal, real-temporal, and real-logical order needles generation pipelines, with 10K/2K/2K train/dev/test splits, and an evaluation model for efficient assessment.

Experiments show Claude, GPT-4o, Gemini, LLaMA, and Qwen struggle with the benchmark, revealing its complexity and the need for model improvements. Noise analysis confirms its reliable and challenging, marking a valuable contribution to the NLP community.

Limitations

Model evaluations may be biased by the dataset's domain, and unoptimized API parameters could affect performance and fairness. Addressing these is crucial for accurate assessments.

The dataset is for academic and research use only; commercial or misuse is prohibited to maintain integrity and ethical standards.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2024. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Chenxin An, Shansan Gong, Ming Zhong, Xingjian Zhao, 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*.

Anthropic. 2024. Introducing the next generation of claude 3.5. https://www.anthropic.com/news/claude-3-5-sonnet.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Aydar Bulatov, Yuri Kuratov, Yermek Kapushev, and Mikhail S. Burtsev. 2024. Scaling transformer to 1m tokens and beyond with rmt. *Preprint*, arXiv:2304.11062.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending Context Window of Large Language Models via Positional Interpolation. *Preprint*, arXiv:2306.15595.

Daniel Y Fu, Elliot L Epstein, Eric Nguyen, Armin W Thomas, Michael Zhang, Tri Dao, Atri Rudra, and Christopher Ré. 2023a. Simple hardware-efficient long convolutions for sequence modeling. In *International Conference on Machine Learning*, pages 10373–10391. PMLR.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023b. Gptscore: Evaluate as you desire. *Preprint*, arXiv:2302.04166.

Mingqi Gao, Xinyu Hu, Jie Ruan, Xiao Pu, and Xiaojun Wan. 2024. Llm-based nlg evaluation: Current status and challenges. *Preprint*, arXiv:2402.01383.

Mingqi Gao, Jie Ruan, Renliang Sun, Xunjian Yin, Shiping Yang, and Xiaojun Wan. 2023. Human-like summarization evaluation with chatgpt. *Preprint*, arXiv:2304.02554.

Alberto García-Durán, Sebastijan Dumančić, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. *arXiv preprint arXiv:1809.03202*.

Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, and 1 others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

gkamradt. 2023. Needle in a haystack - pressure testing llms.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces. *Preprint*, arXiv:2312.00752.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*.

Qi Jia, Siyu Ren, Yizhu Liu, and Kenny Q. Zhu. 2023. Zero-shot faithfulness evaluation for text summarization with foundation language model. *Preprint*, arXiv:2310.11648.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020. Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. *arXiv preprint arXiv:1904.05530*.

Pei Ke, Bosi Wen, Zhuoer Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, Jie Tang, and Minlie Huang. 2024. Critiquellm: Towards an informative critique generation model for evaluation of large language model generation. *Preprint*, arXiv:2311.18702.

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*.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2024a. Loogle: Can long-context language models understand long contexts? *arXiv preprint arXiv:2311.04939*, arXiv:2311.04939.

Mo Li, Songyang Zhang, Yunxin Liu, and Kai Chen. 2024b. Needlebench: Can llms do retrieval and reasoning in 1 million context window? *arXiv preprint arXiv:2407.11963*.

Qintong Li, Leyang Cui, Lingpeng Kong, and Wei Bi. 2023. Collaborative evaluation: Exploring the synergy of large language models and humans for open-ended generation evaluation. *Preprint*, arXiv:2310.19740.

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, and 1 others. 2024a. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*.

Jiaheng Liu, Zhiqi Bai, Yuanxing Zhang, Chenchen Zhang, Yu Zhang, Ge Zhang, Jiakai Wang, Haoran Que, Yukang Chen, Wenbo Su, Tiezheng Ge, Jie Fu, Wenhu Chen, and Bo Zheng. 2024b. E^2llm: Efficient and extreme length extension of large language models. *Preprint*, arXiv:2401.06951.

liuhuanyong. 2022. Finance event graph.

Adian Liusie, Potsawee Manakul, and Mark J. F. Gales. 2024. Llm comparative assessment: Zero-shot nlg evaluation through pairwise comparisons using large language models. *Preprint*, arXiv:2307.07889.

Z Luo, Q Xie, and S Ananiadou. 2023. Chatgpt as a factual inconsistency evaluator for abstractive text summarization (2023). *arXiv preprint arXiv:2303.15621*.

Moonshot. Kimi. `https://www.moonshot.cn`.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, Xuzheng He, Haowen Hou, Jiaju Lin, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, and 15 others. 2023a. Rwkv: Reinventing rnns for the transformer era. *Preprint*, arXiv:2305.13048.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023b. Yarn: Efficient context window extension of large language models. *Preprint*, arXiv:2309.00071.

Ofir Press, Noah Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*.

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*.

Mingyang Song, Mao Zheng, and Xuan Luo. 2024. Counting-stars: A multi-evidence, position-aware, and scalable benchmark for evaluating long-context large language models. *arXiv preprint arXiv:2403.11802*.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063.

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

Qingyang Wu, Zhenzhong Lan, Kun Qian, Jing Gu, Alborz Geramifard, and Zhou Yu. 2022. Memformer: A memory-augmented transformer for sequence modeling. *Preprint*, arXiv:2010.06891.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, Madian Khabsa, Han Fang, Yashar Mehdad, Sharan Narang, Kshitiz Malik, Angela Fan, Shruti Bhosale, Sergey Edunov, Mike Lewis, and 2 others. 2023. Effective long-context scaling of foundation models. *Preprint*, arXiv:2309.16039.

Wenda Xu, Danqing Wang, Liangming Pan, Zhenqiao Song, Markus Freitag, William Yang Wang, and Lei Li. 2023. Instructscore: Explainable text generation evaluation with finegrained feedback. *Preprint*, arXiv:2305.14282.

yuyijiong. 2023. LongData-Corpus. https://huggingface.co/datasets/yuyijiong/LongData-Corpus. Accessed: 2023-12-20.

Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, and 1 others. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. 2024a. Long Context Compression with Activation Beacon. *Preprint*, arXiv:2401.03462.

Qian-Wen Zhang, Fang Li, Jie Wang, Lingfeng Qiao, Yifei Yu, Di Yin, and Xing Sun. 2025. Factguard: Leveraging multi-agent systems to generate answerable and unanswerable questions for enhanced long-context llm extraction. *arXiv preprint arXiv:2504.05607*.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024b. ∞bench: Extending long context evaluation beyond 100k tokens. *arXiv preprint arXiv:2402.13718*.

Yangjun Zhang, Pengjie Ren, and Maarten de Rijke. 2021. A human-machine collaborative framework for evaluating malevolence in dialogues. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5612–5623, Online. Association for Computational Linguistics.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. 2023. Judgelm: Fine-tuned large language models are scalable judges. *Preprint*, arXiv:2310.17631.

A Distribution of the number of needles at different text lengths

We present the distribution of the number of needles at different text lengths of the test set in Table 6. We not only list the distribution of the number of needles in the entire test set, but also provide the distribution of needles across different text length intervals. Both the overall distribution and the grouped distribution maintain consistency.

# needles	# QA	Proportion	8k-16k	16k-32k	32k-64k	64k-128k
3-5	619	30.95%	79 (36%)	195 (32%)	187 (30%)	158 (28%)
6-10	1,028	51.40%	110 (50%)	309 (50%)	304 (50%)	305 (54%)
>10	353	17.65%	31 (14%)	106 (17%)	119 (20%)	97 (18%)
Total	2,000	100%	220	610	610	560

Table 6: Distribution of the number of needles.

B Detailed Performance of Evaluation Model

Table 7 presents the detailed performance of our evaluation model. Two critical points should be noted:

- For the GT (Ground Truth) answer group, when the question doesn't require ordered output, all candidate answers should be judged as "correct". When ordered output is required, "w/ shuffle" answers should be judged as "incorrect" and "w/o shuffle" answers should be judged as "correct". Moreover, All non-GT group answers should be judged as "incorrect".

- Only temporal-order needles require grouping based on question requirements. Logical-order needles must always be output sequentially, thus requiring no question-based grouping.

C Failure modes analysis

The evaluation model we trained not only determines whether an answer is correct, but also provides the rationale behind its judgment. This allows us to analyze the primary sources of failure modes based on these diagnostic explanations. Tabel 8 shows our findings.

We analyzed the proportion of different types of failure modes in the erroneous samples of

Claude-3.5									
Needle types	Question	Answer groups							
		GT		Missing		Redundant		Incorrect	
		w/o shuffle	w/ shuffle	w/o shuffle	w/ shuffle	w/o shuffle	w/ shuffle	w/o shuffle	w/ shuffle
Temporal order	w/ order	1 / 74	58 / 8	64 / 7	73 / 2	56 / 25	74 / 5	59 / 2	70 / 3
	w/o order	0 / 48	27 / 25	42 / 4	41 / 2	36 / 19	40 / 10	42 / 2	47 / 4
Logical order	w/ order	12 / 107	90 / 20	107 / 7	125 / 10	113 / 17	114 / 22	92 / 15	110 / 29
GPT-4o									
Needle types	Question	Answer groups							
		GT		Missing		Redundant		Incorrect	
		w/o shuffle	w/ shuffle	w/o shuffle	w/ shuffle	w/o shuffle	w/ shuffle	w/o shuffle	w/ shuffle
Temporal order	w/ order	0 / 75	59 / 7	67 / 4	75 / 0	68 / 13	72 / 7	61 / 0	72 / 1
	w/o order	0 / 48	15 / 37	45 / 1	42 / 1	47 / 8	39 / 11	44 / 0	46 / 5
Logical order	w/ order	0 / 119	108 / 2	114 / 0	135 / 0	129 / 1	135 / 1	107 / 0	139 / 0
Our Evaluation Model									
Needle types	Question	Answer groups							
		GT		Missing		Redundant		Incorrect	
		w/o shuffle	w/ shuffle	w/o shuffle	w/ shuffle	w/o shuffle	w/ shuffle	w/o shuffle	w/ shuffle
Temporal order	w/ order	0 / 75	64 / 2	71 / 0	75 / 0	81 / 0	79 / 0	61 / 0	73 / 0
	w/o order	0 / 48	8 / 44	46 / 0	43 / 0	55 / 0	50 / 0	44 / 0	51 / 0
Logical order	w/ order	0 / 119	110 / 0	114 / 0	135 / 0	130 / 0	136 / 0	107 / 0	139 / 0

Table 7: Detailed performance of evaluation model compared with Claude-3.5 and GPT-4o on synthetic test data. The 'w/ order' vs. 'w/o order' indicates whether the question requires LLMs to output results in temporal order. The 'w/ shuffle' vs. 'w/o shuffle' condition determines whether the answer candidates were shuffled before evaluation. The 'A/B' ratio reflects [incorrect/correct] judgments obtained by LLMs within the test subset. The red font indicates the number of misjudged samples that deviate from expected results.

Model	Missing items	Redundant items	Incorrect items	Wrong order	Others	No. incorrect samples
Claude-3.5	851 (77.58%)	177 (16.13%)	222 (20.24%)	276 (25.16%)	149 (13.58%)	1097
GPT-4o	956 (66.71%)	338 (23.59%)	249 (17.38%)	189 (13.19%)	936 (65.32%)	1433
GPT-4o-mini	1111 (79.70%)	166 (11.91%)	269 (19.30%)	318 (22.81%)	135 (9.68%)	1394
Gemini-1.5	471 (64.52%)	99 (13.56%)	112 (15.34%)	171 (23.42%)	118 (16.16%)	730
LLaMA-3.3	880 (73.03%)	112 (9.29%)	262 (21.74%)	371 (30.79%)	29 (2.41%)	1205
Qwen-2.5-72B	724 (72.47%)	128 (12.81%)	253 (25.33%)	283 (28.33%)	10 (1.00%)	999
Qwen-2.5-32B	919 (75.89%)	136 (11.23%)	303 (25.02%)	391 (32.29%)	35 (2.89%)	1211
Qwen-2.5-7B	994 (76.23%)	192 (14.72%)	331 (25.38%)	395 (30.29%)	28 (2.15%)	1304
Qwen-3-32B	814 (73.60%)	193 (17.45%)	289 (26.13%)	322 (29.11%)	28 (2.53%)	1106
Qwen-3-14B	912 (73.14%)	253 (20.29%)	306 (24.54%)	402 (32.24%)	35 (2.81%)	1247
Qwen-3-8B	883 (63.85%)	401 (28.99%)	536 (38.76%)	433 (31.31%)	60 (4.34%)	1383

Table 8: Failure modes analysis of different LLMs. All incorrectly evaluated samples were categorized by failure type, and the percentage for each category is provided.

each LLM based on benchmark evaluation results. Please note that the same error sample may contain more than one error pattern. Therefore, the sum of the percentages in each row of the table is greater than 1. we can find that Missing answer items is the most prevalent failure mode. "Others" generally refers to cases where the output is irrelevant to the question or contains additional content. Notably, the Qwen series of models exhibit a lower frequency of this error pattern, suggesting that they have a stronger ability to understand the question in long-context scenarios.

D Noise Analysis Across Groups

Figure 7 presents the result consistency metric of the LLMs' responses across all noise test groups, organized by different text lengths and numbers of needles. The data shows that variations in result consistency across different text lengths are minimal, suggesting that the complexity of test samples constructed by this benchmark is largely uniform across various text lengths. However, as text length increases, there is a rise in the proportion of consistently wrong answers. This trend indicates that the task becomes more challenging with longer texts, making it more difficult to maintain model accuracy by adjusting the needles. Similarly, when the number of needles is 10 or fewer, the variation in re-

sult consistency remains small. However, when the number of needles surpasses 10, there is a marked increase in result consistency. This rise is primarily due to the higher task difficulty associated with a larger number of needles, leading to a corresponding increase in consistently wrong answers, which aligns with expectations.

E Noise Example

Figure 8 provides a schematic diagram illustrating the addition of different types of noise to the raw test data.

F Computational resources and time requirements

F.1 Inference stage

For closed-source models, the main cost comes from API calls, requiring about 97 million tokens for a full test set inference. Response time is 4-7 seconds per request, and multi-threading can improve efficiency if API bandwidth allows.

For open-source models, inference is deployed on a server with 8 H20 GPUs, with response times ranging from 3-5 seconds per request, depending on the model size. Detailed stats will be available in the appendix.

F.2 Evaluation stage

The evaluation model is fine-tuned on Qwen2.5-Instruct-32B and deployed on a server with 8 H20 GPUs. Response time is about 0.5 seconds per request, and a full evaluation takes around 16 minutes, faster with multi-threading.

G Prompts

G.1 Sequential-NIAH data example

In Figure 9, we provide prompts for building a Sequential-NIAH data.

G.2 Prompts for evaluation model

In Figure 10, we provide the prompt to obtain the R_i for building the training data of our evaluation model, which is the prompt for reason analysis in Figure 3.

In Figure 11, we provide the prompt for evaluating the response of LLMs for temporal-order needles extraction.

In Figure 12, we provide the prompt for evaluating the response of LLMs for logical-order needles extraction.

G.3 Prompts for synthetic-temporal order needles

In Figure 13, we provide the prompt of QA template of Chinese synthetic-temporal order needles.

In Figure 14, we provide the prompt of QA template of English synthetic-temporal order needles.

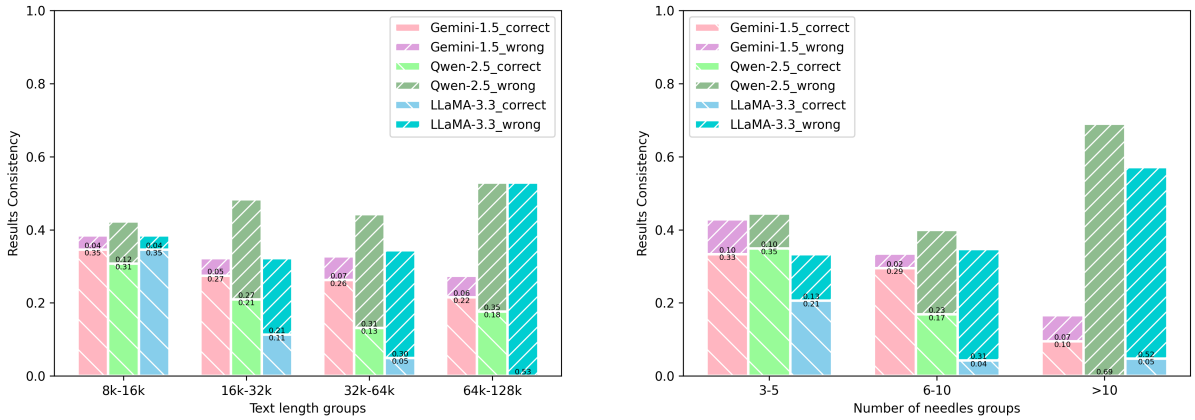
G.4 Prompts for Real-temporal order needles

In Figure 15, we provide the prompt of QA template of English synthetic-temporal order needles.

G.5 Prompts for Real-logical order needles

In Figure 16, we provide the prompt of QA screening from QA pool.

In Figure 17, we provide the prompt of needles generation based on answer items.



(a) Results across length groups.

(b) Results across the number of needles groups.

Figure 7: Noise analysis results of LLMs on noise test data across different groups with correct consistency and wrong consistency.

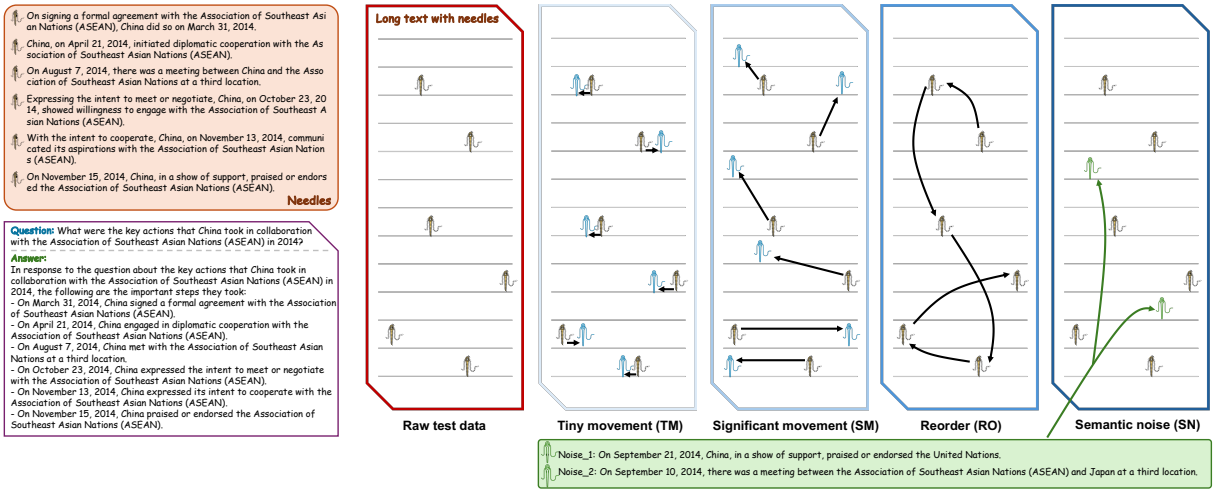


Figure 8: Four types of noise added into the raw test data.

Prompt Template

```
# question: The question of a QA pair
# text_with_needles: The long text inserted by needles
# input_content: The query
# output: The reference answer (The GT answer of a QA pair)

if lang == 'zh':
    Q_choices = ["问题：","问题：\n","问题:","问题:\n","根据下面的文档回答问题：\n"]
    D_choices = ["文档：","文档：\n","文档:","文档:\n"]
    input_content = random.choice(Q_choices) + question + "\n\n" + random.choice(D_choices) +
text_with_needles

elif lang == 'en':
    Q_choices = ["Question: ","Question: \n","Question:","Question:\n","Answer the questions
according to the following documents:\n"]
    D_choices = ["Document: ","Document: \n","Document:","Document:\n"]
    input_content = random.choice() + question + "\n\n" + random.choice() + text_with_needles

Output = answer
```

Figure 9: Prompt to build a Sequential-NIAH data.

Prompt Template (Chinese)

你将会被提供一个问题以及对应的两条答案。在这两条答案中，第一条是待评估答案，第二条是标准答案。以标准答案为参考，待评估答案的回答是否正确已经给出（“正确”或“错误”）。
你需要根据给出的结论来分析原因并详细阐述原因，分析原因时请同时分析问题是否要求按时间顺序回答。
不要让问题以及两条答案的长度影响你的分析。
尽可能保持客观。请严格遵循以下格式输出最终结论。
如果给出的结论是“正确”，请输出“正确”并分析原因。
如果给出的结论是“错误”，请输出“错误”并分析原因。
下面是本次需要评判的内容：
<问题开始>
{query}
<问题结束>
<待评估答案开始>
{answer}
<待评估答案结束>
<标准答案开始>
{ref_answer}
<标准答案结束>
<已知结论开始>
{result}
<已知结论结束>
结果用json形式表示：

```json
{
"order_analysis": <分析问题是否要求按时间顺序回答>,
"eval_analysis": <分析过程>,
}
```

Prompt Template (English)

You will be provided with a question and two corresponding answers. Among these two answers, the first is the answer to be evaluated, and the second is the standard answer. Based on the standard answer, the correctness of the answer to be evaluated has already been given ("correct" or "incorrect").
You need to analyze the reason for the given conclusion and elaborate on it in detail. When analyzing the reason, please also consider whether the question requires a response in temporal order.
Do not let the length of the question or the two answers influence your analysis.
Strive to remain as objective as possible. Please strictly adhere to the following format for the final output.
If the given conclusion is "correct," output "correct" and analyze the reason.
If the given conclusion is "incorrect," output "incorrect" and analyze the reason.
Below is the content to be evaluated this time:
<query_start>
{query}
<query_end>
<answer_start>
{answer}
<answer_end>
< ref_answer_start>
{ref_answer}
< ref_answer_end>
<result_start>
{result}
<result_end>
Present the result in JSON format:

```
{
    "order_analysis": <Analyze whether the question requires a response in temporal order>,
    "eval_analysis": <Analysis process>,
}
```

Figure 10: Prompt for reason analysis to build the training data of evaluation model.

Prompt Template (Chinese)

你将会被提供一个问题以及对应的两条答案。在这两条答案中，第一条是待评估答案，第二条是标准答案。你需要以标准答案为参考，判断待评估答案的回答是否正确。
请以公正的评判者身份评估待评估答案与标准答案，分析原因时请同时分析问题是否要求按时间顺序回答。
不要让问题以及两条答案的长度影响你的评估。
尽可能保持客观。请严格遵循以下格式输出最终结论。
如果待评估答案最终得出的结论与标准答案的最终结论一致且符合问题要求，请回复“正确”。
如果待评估答案最终得出的结论与标准答案的最终结论不一致或不符合问题要求，请输出“错误”。
下面是本次需要评判的内容：
<问题开始>
{query}
<问题结束>
<待评估答案开始>
{answer}
<待评估答案结束>
<标准答案开始>
{ref_answer}
<标准答案结束>
结果用json形式表示：

```json
{
“order_analysis”：<分析问题是否要求按时间顺序回答>,
"eval_analysis"：<评估的分析过程>,
"eval_label"：<最终的评估结果-正确或错误>
}
```

Prompt Template (English)

You will be provided with a question and two corresponding answers. Among these two answers, the first is the answer to be evaluated, and the second is the standard answer. You need to use the standard answer as a reference to determine whether the answer to be evaluated is correct.
Please act as a fair judge to evaluate the answer to be evaluated against the standard answer. When analyzing the reasons, also consider whether the question requires a response in chronological order.
Do not let the length of the question or the two answers influence your evaluation.
Strive to remain as objective as possible. Please strictly adhere to the following format for the final output.
If the conclusion of the answer to be evaluated matches the final conclusion of the standard answer and meets the requirements of the question, reply with "correct."
If the conclusion of the answer to be evaluated does not match the final conclusion of the standard answer or does not meet the requirements of the question, output "incorrect."
Below is the content to be evaluated this time:
<query_start>
{query}
<query_end>
<answer_start>
{answer}
<answer_end>
< ref_answer_start>
{ref_answer}
< ref_answer_end>
Present the result in JSON format:

```
{
    "order_analysis": <Analyze whether the question requires a response in chronological order>,
    "eval_analysis": <Evaluation analysis process>,
    "eval_label": <Final evaluation result—correct or incorrect>
}
```

Figure 11: Prompt for evaluating the response of LLMs for temporal-order needles extraction.

Prompt Template (Chinese)

你将会被提供一个问题以及对应的两条答案。在这两条答案中，第一条是待评估答案，第二条是标准答案。你需要以标准答案为参考，判断待评估答案的回答是否正确。
请以公正的评判者身份评估待评估答案与标准答案，分析原因时需要注意待评估答案的顺序和标准答案是否一致，只有完全一致才可视为正确。
不要让问题以及两条答案的长度影响你的评估。
尽可能保持客观。请严格遵循以下格式输出最终结论。
如果待评估答案与标准答案一致且符合问题要求，请回复“正确”。
如果待评估答案与标准答案不一致或不符合问题要求，请输出“错误”。
下面是本次需要评判的内容：
<问题开始>
{query}
<问题结束>
<待评估答案开始>
{answer}
<待评估答案结束>
<标准答案开始>
{ref_answer}
<标准答案结束>

结果用json形式表示：
```json
{
"eval_analysis": <评估的分析过程>,
"eval_label": <最终的评估结果-正确或错误>
}
```

Prompt Template (English)

You will be provided with a question and two corresponding answers. Among these two answers, the first is the answer to be evaluated, and the second is the standard answer. You need to use the standard answer as a reference to determine whether the answer to be evaluated is correct.
Please act as a fair judge to evaluate the answer to be evaluated against the standard answer. When analyzing the reasons, pay attention to whether the order of the answer to be evaluated matches the standard answer—only a complete match can be considered correct.
Do not let the length of the question or the two answers influence your evaluation.
Strive to remain as objective as possible. Please strictly adhere to the following format for the final output.
If the answer to be evaluated matches the standard answer and meets the requirements of the question, reply with "correct."
If the answer to be evaluated does not match the standard answer or does not meet the requirements of the question, output "incorrect."
Below is the content to be evaluated this time:
<query_start>
{query}
<query_end>
<answer_start>
{answer}
<answer_end>
< ref_answer_start>
{ref_answer}
< ref_answer_end>
Present the result in JSON format:
```
{
    "eval_analysis": <Evaluation analysis process>,
    "eval_label": <Final evaluation result—correct or incorrect>
}
```

Figure 12: Prompt for evaluating the response of LLMs for logical-order needles extraction.

Prompt Template (Chinese)

```
zh_QA_template = [
    ["按顺序列出文档中{}的具体内容有哪些？", "{}的具体内容如下：",
    ["请按顺序提供{}的相关信息", "{}的相关信息如下：",
    ["罗列{}的内容都有什么？", "{}的内容包括：",
    ["文中提到的{}都包括什么内容？", "根据文中提及信息，{}的内容包括：",
    ["{}记录了哪些事件？", "{}对以下事件进行了记录：",
    ["文档里的{}提到了哪些事？请按日期罗列事件", "文档里的{}提到了以下事件：",
    ["按日期罗列{}的内容", "按日期罗列{}的内容如下：",
    ["请按日期罗列文章中提到的{}内容。", "根据文章所述，按日期罗列{}的内容如下：",
    ["请按时间顺序给出{}里作者都做了些什么", "根据{}的时间顺序，作者所经历的活动如下：",
    ["请按时间顺序列举{}里作者都参与了哪些事？", "根据提供的文档内容，以下是{}的作者按时间顺序所参与的事项：
    "],
    ["列举{}在{}都做了哪些事", "{}做了以下事情：",
    ["按时间顺序列举{}在{}期间干了什么？", "{}在{}期间做了以下事情：",
    ["分析文档，按时间顺序列举{}在{}期间干了什么？", "基于文档分析，{}在{}期间依次做了以下事情：",
    ["按顺序列举{}在{}期间的活动内容？", "{}在{}期间的活动内容如下：",
    ["基于文档，梳理一下{}在{}的活动内容有哪些？", "基于文档，我找到了{}在{}期间的活动内容如下：",
    ["根据上述文档，按时间顺序输出{}在{}记录的信息？", "根据上述文档，{}在{}期间记录的信息如下："]
]
```

Figure 13: Prompt for QA template of Chinese synthetic-temporal order needles.

Prompt Template (English)

```
en_QA_template = [
    ["List the specific contents of {} in order.", "The specific contents of {} are as follows:"],
    ["Please provide the relevant information of {} in order", "The relevant information of {} is as follows:"],
    ["What are the contents of {}?", "The contents of {} include:"],
    ["What contents are mentioned in the text about {}?", "According to the information mentioned in the text, the contents of {} include:"],
    ["What events are recorded in {}?", "{} records the following events:"],
    ["What events are mentioned in the {}? Please list the events by date", "The {} mentions the following events:"],
    ["List the contents of {} by date", "The contents of {} by date are as follows:"],
    ["Please list the contents of {} mentioned in the article by date.", "According to the article, the contents of {} by date are as follows:"],
    ["What did the author do in {} in chronological order", "According to the chronological order of {}, the activities experienced by the author are as follows:"],
    ["What events did the author participate in {} in chronological order?", "Based on the provided document content, the following are the events the author
    participated in {} in chronological order:"],
    ["List what {} did in {}", "{} did the following things:"],
    ["List what {} did during {} in chronological order?", "{} did the following things during {}:"],
    ["Analyze the document and list what {} did during {} in chronological order?", "Based on the document analysis, {} did the following things during {} in order:"],
    ["List the activities of {} during {} in chronological order?", "The activities of {} during {} are as follows:"],
    ["Based on the document, organize the activities of {} during {}?", "Based on the document, I found the following activities of {} during {}:"],
    ["Based on the above document, output the information recorded by {} during {} in chronological order?", "Based on the above document, the information
    recorded by {} during {} is as follows:"]
]
```

Figure 14: Prompt for QA template of English synthetic-temporal order needles.

Pipeline and prompts

```
def gen_needle(relation):

    template = " Given the following information:\n\n <Information Start>\n{}\n<Information End>\n\n" + \
    "Please describe the event mentioned in the information in one sentence. If some content is 'N/A', you may ignore it." + \
    "Please output this sentence: "
    content = template.format('\n'.join([f"{k}:{v}" for k, v in relation.items()]))
    res_txt = return_by_llm(content)
    return res_txt

def gen_Q(needles):

    template = " Given the following information:\n- {}\nPlease propose a question related to each item of information, and output the question directly: "
    content = template.format('\n- '.join(needles))
    res_txt = return_by_llm(content)
    if ':' in res_txt:
        res_txt = ''.join(res_txt.split(':')[1:]).strip()
    return res_txt


def gen_A(question, needles):

    template = " Known question: {}\nKnown answer items: {}\nPlease add a simple connecting sentence before all answer items to ensure semantic
    coherence with them. Do not modify the numbering format of the answer items. Please output the revised answers: "
    content = template.format(question, '\n- '.join(needles))
    res_txt = return_by_llm(content)
    return res_txt
```

Figure 15: Pipeline and prompts for needles and QA generation of real-temporal order needles.

Prompts

You will be provided with a set of Q&A items containing numbered entries. Please determine whether these answers follow a rigorous logical order.

<Question Start>
QQQ
<Question End>

<Answer Start>
AAA
<Answer End>

Please determine whether these numbered answers follow a rigorous logical order:
If shuffling these items does not affect correctness, it indicates no logical order. Answer "No."
If shuffling these items affects correctness, it indicates a logical order. Answer "Yes."
Present the result in JSON format:

```json
{
   "reason": <analysis reason>,
   "is_right": <judgment result - Yes or No>
}
```

Figure 16: Prompt for QA screening from QA pool.

Prompts

Given a question and several answer key points related to it, please rewrite these key points so that each one includes the information from the question.\n\n" \
    "<Question Start>\n{}\n<Question End>\n\n" \
    "<Answer Key Points Start>\n{}\n<Answer Key Points End>\n\n" \
    "Rewritten Answer Key Points:\n

Figure 17: Prompt for needles generation based on answer items.