Counting-Stars (★): A Multi-evidence, Position-aware, and Scalable Benchmark for Evaluating Long-Context Large Language Models

Mingyang Song, Mao Zheng, Xuan Luo
Tencent Hunyuan
nickmysong@tencent.com

Abstract

Despite recent efforts to develop large language models with robust long-context capabilities, the lack of long-context benchmarks means that relatively little is known about their performance. To alleviate this gap, in this paper, we propose **Counting-Stars**, a multi-evidence, position-aware, and scalable benchmark designed to evaluate the multi-evidence retrieval capabilities of long-context LLMs. **Counting-Stars** comprises two counting-based multiple pieces of evidence retrieval sub-tasks: searching and reasoning. Using Counting-Stars, we conduct experiments to evaluate several long-context LLMs, including GPT-4 Turbo, Gemini 1.5 Pro, Claude3 Opus, GLM-4, and Moonshot-v1. Extensive experimental results demonstrate that Gemini 1.5 Pro achieves the best overall results, while GPT-4 Turbo exhibits the most stable performance across various tasks. Furthermore, our analysis of these LLMs, which have been extended to handle long-context scenarios, indicates that significant room for improvement remains as the length of the input context and the complexity of the tasks increase. The code and data are publicly accessible here[1].

1 Introduction

Large language models (LLMs) have demonstrated exceptional performance across a wide range of Natural Language Processing (NLP) downstream tasks (Huang et al., 2023). A context window of 128K tokens is crucial for LLMs and enables LLMs to perform tasks that are significantly beyond the existing paradigm, such as multi-document question answering (Caciularu et al., 2023), repository-level code understanding (Bairi et al., 2023), etc. An increasing number of studies focus on extending the context window these models can handle to enable LLMs to support more intricate and diverse applications. Despite these developments, the efficacy of models in long-context settings still needs to be examined, primarily due to the lack of a robust evaluation benchmark (An et al., 2023; Liu et al., 2023; Fu et al., 2024).

In contrast to the rapid evolution of the supported context length of LLMs, existing benchmarks have lagged behind (Yuan et al., 2024). Meanwhile, it is worth mentioning that tasks in existing benchmarks are primarily short-context tasks, which only require LLMs to find evidence for answering questions within a short context to test the performance of LLMs instead of a long context (Li et al., 2023b; Fu et al., 2024). A few benchmarks have been proposed for evaluating long-context LLMs, including LongBench (Bai et al., 2023b), LooGLE (Li et al., 2023b), ∞Bench (Zhang et al., 2024), which have been instrumental in evaluating the performance of long-context LLMs.

Recently, the needle-in-a-haystack benchmark[2] has become a popular benchmark for evaluating whether LLMs have the capability of acquiring information from long documents. Specifically, this benchmark requires LLMs to precisely retrieve a specific sentence inserted at an arbitrary position within a long context, thereby assessing their capability to search for the sentence in long contexts. However, many newly released long-context LLMs have adopted the needle-in-a-haystack benchmark to evaluate their long-context processing capabilities and have achieved nearly perfect performance. This renders the needle-in-a-haystack benchmark insufficient to distinguish the differences between these models. Not only does this demonstrate the advancements in recent long-context LLMs, but it also indicates that the needle-in-a-haystack benchmark is too simplistic to further test their capabilities. Generally, acquiring information should be the most fundamental capability of long-text LLMs, which is the prerequisite for completing complex

[1] https://github.com/nick7nlp/Counting-Stars

[2] https://github.com/gkamradt/LLMTest_NeedleInAHaystack

Proceedings of the 31st International Conference on Computational Linguistics, pages 3753–3763
January 19–24, 2025. ©2025 Association for Computational Linguistics

tasks. However, existing long-context benchmarks rarely focus on the multi-evidence collection ability of LLMs. Even when they do, the amount of evidence to be collected is relatively small and not proportional to the amount of evidence that a long document should contain.

To mitigate the shortcomings of existing benchmarks, in this paper, we propose a multi-evidence, position-aware, and scalable benchmark for evaluating long-context LLMs, named Counting-Stars. As the name suggests, the Counting-Stars refers to asking LLMs to count the numbers of stars from multiple sentences describing the number of stars counted by the little penguin inserted in the long context and then summarize into a specified answer. Through the Counting-Stars, we expect to evaluate the long context capabilities of multi-evidence searching and multi-evidence reasoning of LLMs. More specifically, the former focuses on testing the capability of LLMs to retrieve evidence at different positions within the long context, which can more clearly reflect the quality of long-context modeling. When collecting evidence, reasoning is often required to ensure that the evidence gathered supports the correct answer to the question. Therefore, the latter evaluates the LLM's ability to filter out noise or incorrect information when retrieving information and the model's reasoning ability at different positions within the long context. Generally speaking, the latter is definitely more challenging than the former. In other words, the former can be treated as making LLMs distinguish between long contexts and inserted sentences (similar to the needle-in-a-haystack benchmark), while the latter involves distinguishing evidence within each inserted sentence.

Experiments show that the tested LLMs can perform well on the Counting-Stars when the context length is below 32K in most cases. However, as the context length increases, the performance of all models declines. However, this decline is not absolute, meaning a model might achieve better results at 120K than at 100K. Generally, Gemini 1.5 Pro achieved the best results on all tasks, and the performance of GPT-4 Turbo is the most stable across all tasks in the Counting-Stars. Although our experiments may not fully support the loss-in-the-middle phenomenon, it can be observed that most LLMs are good at collecting the numbers of stars located at the beginning and end slightly better than those located in the middle of the long context.

2 Counting-Stars (★)

LLMs have shown remarkable performance across diverse NLP tasks but are constrained by their small context window size (short-context). Recently, various studies have expanded the context length to accommodate up to 128K tokens and more (long-context). The main difference between short- and long-context scenarios is that in the latter, LLMs need to process more information at once, which may lead to the loss of key information, resulting in decreased performance. Therefore, in long-context scenarios, the evaluation of LLMs should focus on the capability of LLMs to acquire information and distinguish incorrect information while acquiring that information.

Multi-evidence. In long-context scenarios, answering a question may require gathering substantial evidence from different positions within the lengthy context. Therefore, it is necessary to verify the ability of LLMs to collect a large amount of evidence from a long context all at once. Moreover, to the best of our knowledge, *Counting-Stars is the first long-context benchmark to significantly increase the number of pieces of evidence* (i.e., increase to 32, 64, 128, 256, 512, and even 1024).

Position-aware. In long-context scenarios, a typical bad case is that when the answer to a question appears in different positions of the long context, the performance of LLMs varies greatly, such as the *lost-in-the-middle* phenomenon (Liu et al., 2023). Therefore, when evaluating the long-context LLMs, it is necessary to reveal which specific positions of evidence are missing or reasoning incorrectly through the evaluation results to analyze the problem more precisely and meticulously.

Scalable. As mentioned earlier, developing long-context benchmarks often lags behind the speed of long-context LLMs. At the same time, constructing a long-context benchmark is difficult and expensive, so easy scalability is essential.

In general, the capacity for a long-context LLM to do human textual instructions largely depends on its *multi-evidence searching* ability. Moreover, an indispensable ability of LLMs extends beyond mere essential evidence collection to encompass *reasoning* based on the collected evidence. Therefore, the Counting-Stars mainly evaluates the long-context capability of LLMs from two perspectives, i.e., *long-context multi-evidence searching* and *long-context multi-evidence reasoning*. Expressly, this test can be understood as asking LLMs

Task Name	Test Example
Long-Context Multi-evidence Searching *English Version*	November 2005In the next few years, venture capital funds will find themselves squeezed from four directions. They're already stuck with a seller's market, because of the huge amounts they raised at the end of the Bubble and still haven't invested. This by itself is not the end of the world. In fact, it's just a more extreme version of the norm in the VC business: too much money chasing too few deals.Unfortunately, those few deals now want less and less money, because it's getting so cheap to start a startup ... ***The little penguin counted {number1} ★*** ... Moore's law, which makes hardware geometrically closer to free; the Web, which makes promotion free if you're good; and better languages, which make development a lot cheaper.When we started our startup in 1995, the first three were our biggest expenses. We had to pay $5000 for the Netscape Commerce Server, the only software that then supported secure http connections ... ***The little penguin counted {number2} ★*** ... people throw away computers more powerful than our first server ... . ***On this moonlit and misty night, the little penguin is looking up at the sky and concentrating on counting ★. Please help the little penguin collect the number of ★, for example: "little_penguin": [x, x, x,...]. The summation is not required, and the numbers in [x, x, x,...] represent the counted number of ★ by the little penguin. Only output the results in JSON format without any explanation.***
Long-Context Multi-evidence Reasoning *English Version*	November 2005In the next few years, venture capital funds will find themselves squeezed from four directions. They're already stuck with a seller's market, because of the huge amounts they raised at the end of the Bubble and still haven't invested. This by itself is not the end of the world. In fact, it's just a more extreme version of the norm in the VC business: too much money chasing too few deals.Unfortunately, those few deals now want less and less money, because it's getting so cheap to start a startup ... ***The little penguin counted {wrong number1} ★, but found that a mistake had been made, so the counting was done again, and this time {number1} ★ was counted correctly.*** ... Moore's law, which makes hardware geometrically closer to free; the Web, which makes promotion free if you're good; and better languages, which make development a lot cheaper.When we started our startup in 1995, the first three were our biggest expenses. We had to pay $5000 for the Netscape Commerce Server, the only software that then supported secure http connections ... ***The little penguin counted {wrong number2} ★, but found that a mistake had been made, so the counting was done again, and this time {number2} ★ was counted correctly.*** ... people throw away computers more powerful than our first server .... ***On this moonlit and misty night, the little penguin is looking up at the sky and concentrating on counting ★. Please help the little penguin collect the correct number of ★, for example: "little_penguin": [x, x, x,...]. The summation is not required, and the numbers in [x, x, x,...] represent the correctly counted number of ★ by the little penguin. Only output the results in JSON format without any explanation.***

Table 1: Prompt templates for the two counting tasks in the English version of Counting-Stars.

to find and remember all the sentences in the long text that describe the little penguin counting stars and organize them into a list to return the final answer. All sentences are prior inserted into a long text at the same interval. In addition, the used long text can be any text data that is not related to the sentences describing the little penguin counting stars, such as The Story of the Stone[3] for the Chinese version of the Counting-Stars and Paul Graham Essays[4] for the English version of the Counting-Stars. Next, we introduce the Counting-Stars in detail.

2.1 Long-Context Multi-evidence Searching

Multi-evidence searching refers to the capability of distinguishing and collecting critical information framed within intricate and long textual data, which bottlenecks the performance of LLMs in synthesizing contextualized knowledge to execute various tasks, from answering multi-document questions to executing complex human instructions. Furthermore, maintaining a comprehensive and accurate grasp of the input text becomes increasingly challenging as the context length increases. Therefore, in the Counting-Stars test, the first task is to examine the multi-evidence searching ability of long-context LLMs, as illustrated in Table 1 (named as *Long-Context Multi-evidence Searching*). In multi-evidence searching, all sentences describing the little penguin counting stars are designed as "***The little penguin counted {number1} ★***". Here, *{number1}* indicates the number of stars the little penguin counted. Concretely, we randomly generated all the numbers of stars as *{number1, number2, ...}* because we found that LLMs easily slack off if a sequence of numbers is continuous or regular. In this task, we hope that LLMs collect all the numbers of stars the little penguin counted and list all digits rather than summation.

2.2 Long-Context Multi-evidence Reasoning

In many real-world tasks, when answering questions under a long and intricate context, it is not only necessary to collect multi-evidence information but also to reason and identify each original piece of evidence before acquiring it to avoid collecting wrong evidence. Therefore, in the Counting-Stars test, the second task is to examine the multi-evidence reasoning ability of long-context LLMs, as illustrated in Table 1 (named as *Long-Context Multi-evidence Reasoning*). In multi-evidence reasoning, all sentences describing the little penguin counting stars are designed as "***The little penguin***

[3] *The Story of the Stone*, is an 18th-century Chinese novel authored by Cao Xueqin, considered to be one of the Four Great Classical Novels of Chinese literature.

[4] The English context data used in this paper is similar to the needle-in-a-haystack.

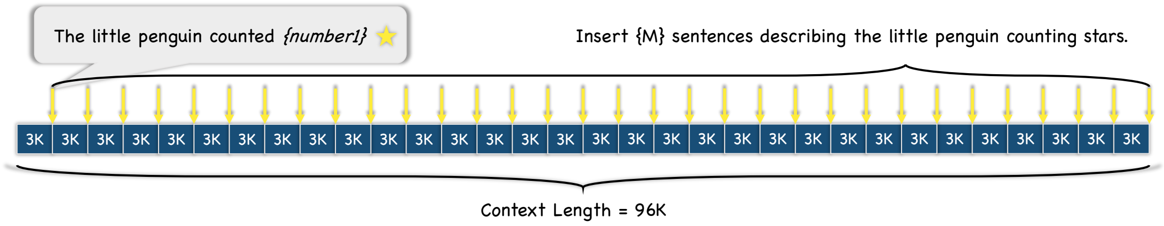

The little penguin counted *{number1}* ★
Insert {M} sentences describing the little penguin counting stars.
3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K 3K
Context Length = 96K

Figure 1: Illustration of how to scatter stars into the long context with the length of 96K.

LLMs	Length Limit	Service Used
GPT-4 Turbo		
gpt4-1106-preview	128K	Accessed from API
gpt4-0125-preview	128K	Accessed from API
Gemini 1.5 Pro	1M	Accessed from poe.com
Claude 3		
Opus	200K	Accessed from poe.com
Sonnet	200K	Accessed from poe.com
Haiku	200K	Accessed from poe.com
GLM-4	128K	Accessed from API
Moonshot-v1	128K	Accessed from API

Table 2: LLMs used in our experiment.

counted {wrong number1} ★, but found that a mistake had been made, so the counting was done again, and this time {number1} ★ was counted correctly.". Here, *{wrong number1}* denotes the number of stars the little penguin counted incorrectly, and *{number1}* indicates the number of stars the little penguin counted correctly. Specifically, *{number1, 2, ...}* are the same as the first task, and *{wrong number1, 2, ...}* are randomly added or subtracted by one based on the *{number1, 2, ...}*. In this task, we hope that LLMs collect all the correct numbers of stars the little penguin counted and summarize them in a list.

2.3 Scalable Test Setting

Various approaches have been proposed to expand the context window of LLMs to accommodate even up to $128K$ input tokens or more. As the length of the context that LLMs accommodate increases, it becomes increasingly difficult to construct a qualified benchmark to evaluate them because the testing length of benchmarks can hardly be arbitrarily scaled in size. In contrast, the testing length of the Counting-Stars test can be set arbitrarily, which can be $128K$, $200K$, or even $1M$. At the same time, the amount of evidence to be collected can also be set arbitrarily. For the number of evidence, we initially set it to $M = 32$, which represents the number of sentences inserted into the long context. It is worth noting that we can also set M to 64, 128, 256, 512, or even 1024. However, we find that when $M = 32$, the Counting-Stars test is already difficult for many LLMs, so this paper only shows the results of each LLM when $M = 32$.

Another parameter that must be specially declared is the number of test samples (N). Similar to the needle-in-a-haystack test, when the context length to be tested is $128K$, it will be tested from $4K$ to $128K$ with $4K$ as the interval for a total of $N = 32$ test data. For example, as shown in Figure 1, when the context length is 96K, it will be tested from $3K$ to $96K$ with $3K$ as the interval for a total of $N = 32$ test data.

3 Experiments

3.1 Baselines and Experimental Settings

In this study, we evaluate the Chinese and English versions of the Counting-Stars test on several famous long-context LLMs that may handle long contexts, including GPT-4 Turbo (OpenAI, 2024), Gemini 1.5 Pro (Reid et al., 2024), Claude 3 Opus[5], GLM-4[6], and Moonshot-v1[7]. Table 2 shows the context length limits (in tokens) of the LLMs GPT-4 Turbo, Gemini 1.5 Pro, Claude 3 Opus, GLM-4, and Moonshot-v1 used in the experiment.

Specifically, in the experiments, we utilize the number of prompt tokens returned by the GPT-4 Turbo API to measure the context length. Therefore, it should also be noted that the position of inserting stars is somewhat biased. Firstly, it is due to the input context length being counted by the number of prompt tokens returned by GPT-4 Turbo. Secondly, it is precisely necessary to ensure some randomness.

Generally, evaluating text-based results is usually more complex, so the evidence to be collected in the Counting-Stars is all numerical, making it

[5] https://www.anthropic.com/news/claude-3-family

[6] https://open.bigmodel.cn/

[7] https://kimi.moonshot.cn/

Models	GPT-4 TURBO		GEMINI 1.5 PRO	CLAUDE3			GLM-4	MOONSHOT-V1
	1106	*0125*		OPUS	SONNET	HAIKU		
P@32								
Multi-evidence Searching (ZH)	0.697	0.663	0.775	0.807	0.788	0.698	0.682	0.606
Multi-evidence Searching (EN)	0.718	0.662	0.833	0.705	-	-	0.389	0.559
P@32*								
Multi-evidence Reasoning (ZH)	0.473	0.386	0.575	0.488	-	-	0.475	0.344
Multi-evidence Reasoning (EN)	0.651	0.610	0.371	0.374	-	-	0.179	0.460
Average Score	0.635$_2$	0.580$_4$	0.639$_1$	0.594$_3$	-	-	0.431$_6$	0.492$_5$

Table 3: The overall performance on the Counting-Stars-(32)-(Multi-evidence Reasoning).

Models	Multi-evidence Searching (ZH)										
	4K	8K	12K	16K	20K	24K	28K	32K	36K∼64K	68K∼96K	100K∼128K
GPT-4 TURBO *(1106)*	1.00	0.97	0.92	0.76	0.84	0.75	0.75	0.78	0.76	0.61	0.57
CLAUDE3 OPUS	1.00	1.00	0.89	0.83	0.86	0.89	0.85	0.78	0.78	0.76	0.80
GEMINI 1.5 PRO	1.00	1.00	0.97	0.91	0.85	0.94	0.61	0.68	0.80	0.74	0.67
GLM-4	1.00	0.86	0.98	0.90	0.96	0.94	0.88	0.92	0.84	0.62	0.37
MOONSHOT-V1	0.94	0.84	0.88	0.88	0.78	0.84	0.41	0.88	0.49	0.55	0.58

Table 4: The P@32 performance on the Chinese version of the Counting-Stars-(32)-(Multi-evidence Searching).

Models	Multi-evidence Searching (EN)				Multi-evidence Reasoning (ZH)				Multi-evidence Reasoning (EN)			
	4K-32K	36K-64K	68K-96K	100K-128K	4K-32K	36K-64K	68K-96K	100K-128K	4K-32K	36K-64K	68K-96K	100K-128K
GPT-4 TURBO *(1106)*	0.88	0.62	0.64	0.74	0.80	0.51	0.28	0.30	0.86	0.62	0.60	0.52
CLAUDE3 OPUS	0.81	0.65	0.65	0.71	0.82	0.52	0.29	0.33	0.72	0.44	0.05	0.29
GEMINI 1.5 PRO	0.90	0.84	0.80	0.78	0.75	0.55	0.50	0.49	0.66	0.24	0.30	0.28
GLM-4	0.57	0.36	0.34	0.29	0.84	0.64	0.25	0.17	0.28	0.09	0.19	0.15
MOONSHOT-V1	0.86	0.43	0.45	0.50	0.65	0.24	0.19	0.29	0.52	0.53	0.44	0.35

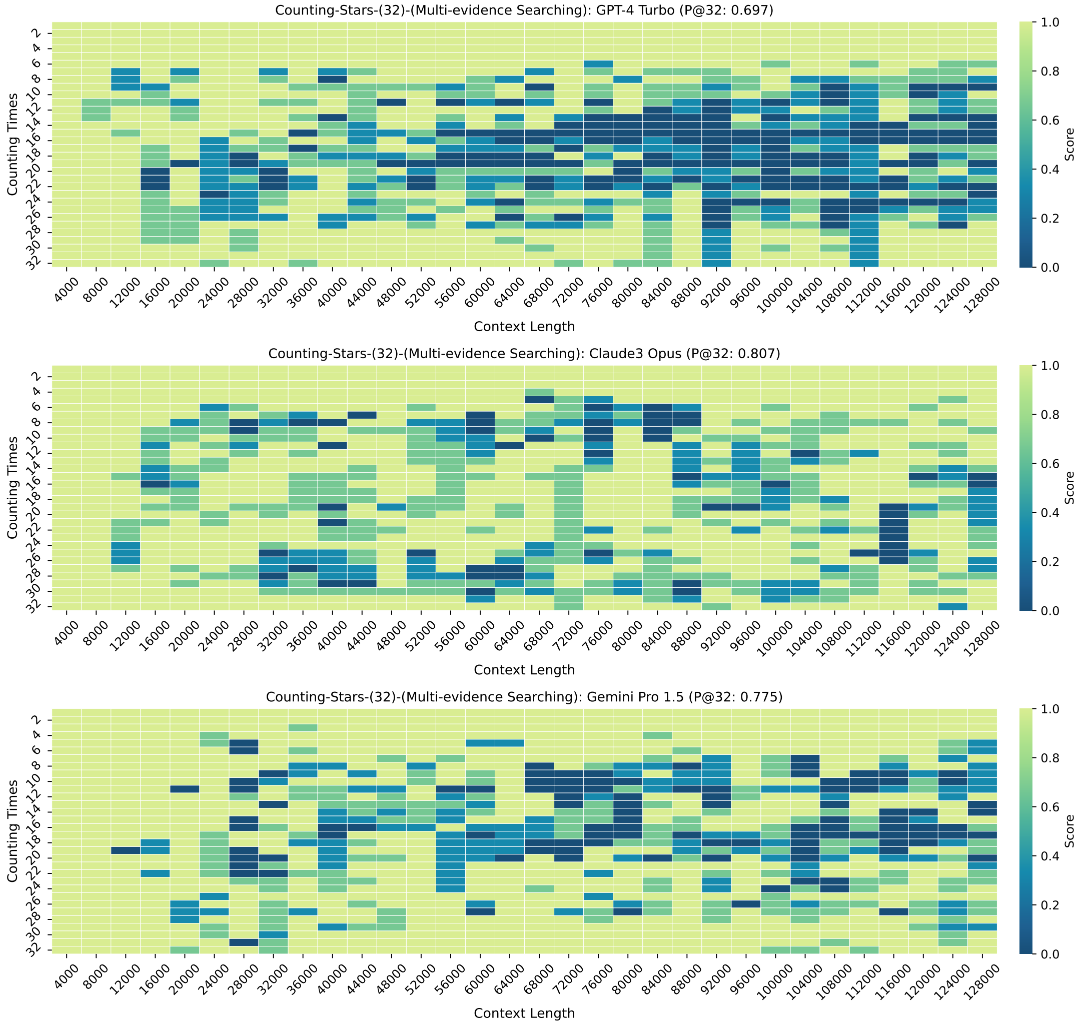
Table 5: The P@32 performance on the English version of the Counting-Stars-(32)-(Multi-evidence Searching) as well as the Chinese and English versions of the Counting-Stars-(32)-(Multi-evidence Reasoning).

more straightforward to evaluate. For a piece of test data, the prediction results are evaluated starting from the first number of stars, that is, *{number1}*, *{number2}*, ..., *{numberM}*. In this paper, we adopt P@N as the evaluation metric, which includes two modes: one where N=M, representing the counting times (here, M denotes the counting times); the other, referencing prior studies (Yuan et al., 2020; Song et al., 2023a,b, 2024), adopts the P@M as the evaluation metric, where M denotes the total number of the retrieved results.

Specifically, for the Multi-evidence Searching task, if the results contain *{number1}*, it gets a score of 1; if it doesn't, it gets 0. Meanwhile, we construct a rule-based evaluation approach for the Multi-evidence Reasoning task, named P@32*. For P@32*, when the retrieved results contain only *{number1}*, the score is 1; if it also contains *{wrong number1}*, the score is 0.5; if the value only contains *{wrong number1}*, the score is 0.25, and if both If not found, the score is 0.

3.2 Overall Performance

Table 7 present the performance of GPT-4 Turbo, Claude3 Opus, Gemini 1.5 Pro, GLM-4, Moonshot-v1 on the Chinese and English versions of the Counting-Stars-(32)-(Multi-evidence Searching) and Counting-Stars-(32)-(Multi-evidence Reasoning). Overall, Claude3 Opus achieves the best performance on the Chinese version of the Counting-Stars-(32)-(Multi-evidence Searching), Gemini 1.5 Pro obtains the best performance on the English version of the Counting-Stars-(32)-(Multi-evidence Searching) and the Chinese version of the Counting-Stars-(32)-(Multi-evidence Reasoning), and GPT-4 Turbo obtains the best performance on the English version of the Counting-Stars-(32)-(Multi-evidence Reasoning). Although these LLMs have achieved nearly perfect performance on the needle-in-a-haystack task, they still perform poorly on the Counting-Stars, which indicates that the needle-in-a-haystack is too simple to truly show the capabilities of LLMs in processing long texts.



Counting-Stars-(32)-(Multi-evidence Searching): GPT-4 Turbo (P@32: 0.697)

Counting-Stars-(32)-(Multi-evidence Searching): Claude3 Opus (P@32: 0.807)

Counting-Stars-(32)-(Multi-evidence Searching): Gemini Pro 1.5 (P@32: 0.775)

Figure 2: Visualization of the results on the Chinese version of the Counting-Stars-32-(Multi-evidence Searching).

The multi-evidence reasoning task necessitates that LLMs engage in acquiring and reasoning multiple pieces of evidence simultaneously, which is more complex than the multi-evidence searching task. This task requires LLMs to sift through and exclude inaccurate evidence while gathering information from a long context to answer questions. As indicated by the data in Table 7, each LLM performs not well enough. Notably, in contrast to GPT-4 Turbo and Claude3 Opus, Gemini 1.5 Pro stands out for having gathered virtually no incorrect information, as shown in Figure 3.

From Table 4 and Table 5, it can be observed that all LLMs are capable of achieving higher scores in short-context scenarios, which confirms that the Counting-Stars is reasonable and can be accomplished by LLMs. However, as the context length increases, the performance of all models shows a downward trend. Among them, GPT-4 Turbo's performance is relatively stable. In addition, GLM-4 has obtained surprising results under the 32K context length of the Chinese version of the Counting-Stars-(32)-(Multi-evidence Searching).

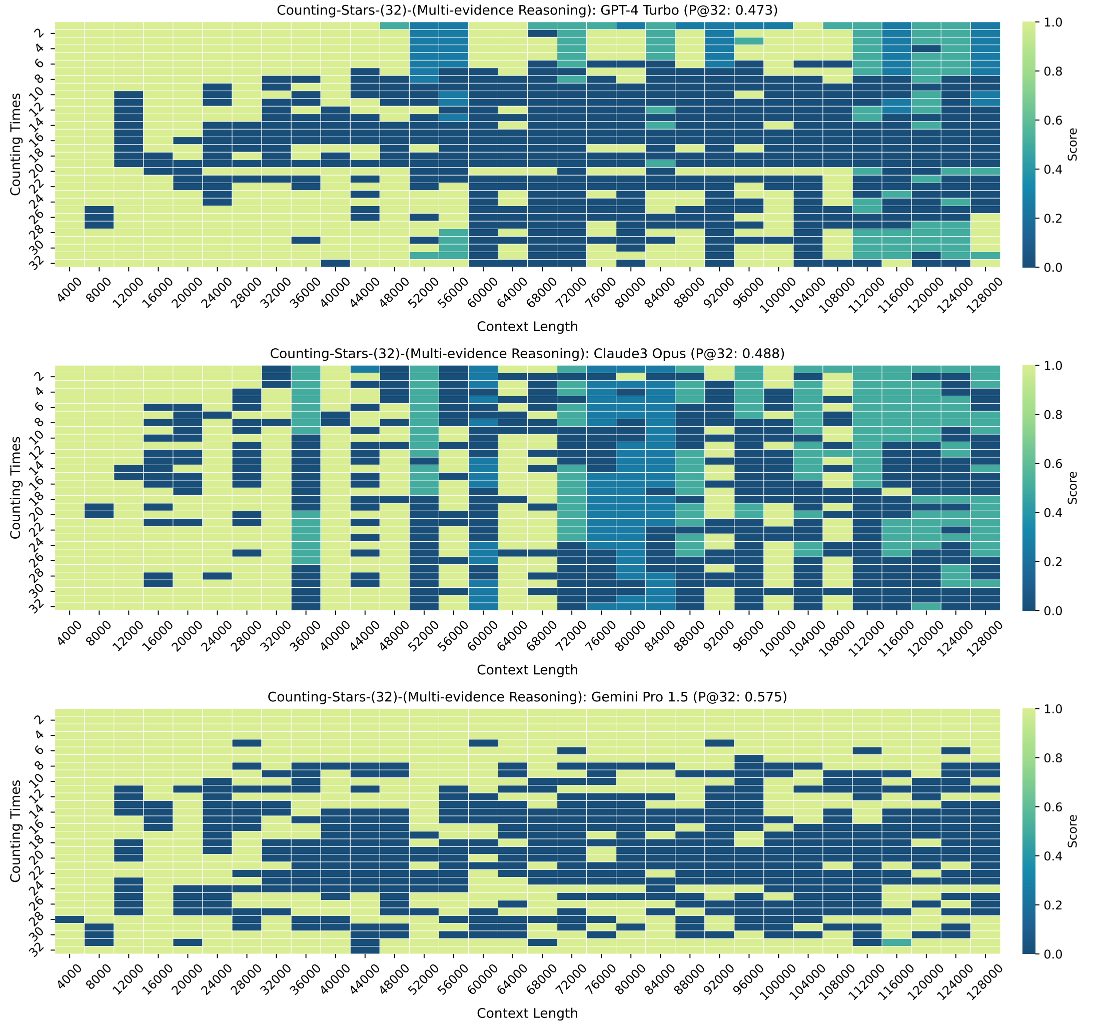
By analyzing the experimental results of several long-context LLMs, we summarize three kinds of bad cases: (1) repeat a single number; (2) generate an increasing array; (3) fail to follow instruction, as shown in Table 6.

4 Discussion

We discuss the length-stability dilemma and the *lost-in-the-middle* phenomenon in this section.

4.1 The Length-Stability Dilemma

One phenomenon that puzzles us the most among the test results of both needle-in-a-haystack and Counting-Stars is why the same task performs well when the input context length is long but badly at the shorter context (e.g., 112K and 108K in Figure 2). It is important to note that this phenomenon


Counting-Stars-(32)-(Multi-evidence Reasoning): GPT-4 Turbo (P@32: 0.473)

Counting-Stars-(32)-(Multi-evidence Reasoning): Claude3 Opus (P@32: 0.488)

Counting-Stars-(32)-(Multi-evidence Reasoning): Gemini Pro 1.5 (P@32: 0.575)

Figure 3: Visualization of the results on the Chinese version of the Counting-Stars-32-(Multi-evidence Reasoning).

becomes more pronounced as the length of the context increases. In other words, hiding the answer in different positions within different contexts results in LLMs failing to search it. Is this due to the different contexts surrounding the answer? Or is it because the distribution of the input context length of the training data is not uniform, leading to differences in the capabilities of LLMs across various context lengths? Therefore, could increase the robustness of LLMs help?

Based on the experiments in this paper, we are not yet able to determine the specific reasons behind this phenomenon; identifying these reasons is a goal that the next version of Counting-Stars aims to achieve. We consider the most intuitive explanation to be that the long-context capabilities of LLMs are still relatively weak, so when resources are limited, some stability must be sacrificed. Addressing this issue could help researchers better analyze and enhance the long-context modeling capabilities of LLMs, benefiting specific NLP tasks such as multi-document question answering. Moreover, stability refers to the understanding and reasoning abilities of LLMs when handling different long contexts, which is more crucial than merely the length of context processing.

4.2 Lost in the Middle

Prior research indicates a performance decline in some LLMs when answers are positioned around the middle of the long context (Liu et al., 2023). Similar to (Zhang et al., 2024), however, our findings can not strongly corroborate the *lost-in-the-middle* phenomenon. One possible reason why we obtain different observations from (Liu et al., 2023) is that they find the phenomenon via the test at most 16K length contexts, which is not long enough. In our experiments based on the Counting-Stars, we discover that the bad cases may not mainly appear in the middle of the long context, especially for

Bad Case Description	**Example**
repeat a single number	[15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, ...]
generate increasing an array	[5, 9, 15, 19, 29, 39, 45, 49, 53, 59, 63, 67, 71, 75, 79, 83, 87, 91, 95, 99, 103, 107, 111, 115, 119, 123, 127, 131, 135, 139, 143, 147, 151, 155, 159, 163, 167, 171, 175, 179, 183, 187, 191, 195, 199, 203, 207, 211, 215, 219, 223, 227, 231, 235, 239, 243, 247, 251, 255, 259, 263, 267, 271, 275, 279, 283, 287, 291, 295, 299, 303, 307, 311, 315, 319, 323, 327, 331, 335, 339, 343, 347, 351, ...]
fail to follow instruction	"The little penguin counted 15 ★", "The little penguin counted 117 ★", "The little penguin counted 42 ★", "The little penguin counted 69 ★", "The little penguins counted 58 ★", "The little penguin counted 107 ★", "The little penguin counted 9 ★", "The little penguin counted 49 ★", "The little penguin counted 113 ★", . . .

Table 6: Bad cases generated by LLMs.

Models	GPT-4 Turbo 1106		Gemini 1.5 Pro		Claude3 Opus	
	F1@32	*F1@M*	*F1@32*	*F1@M*	*F1@32*	*F1@M*
Multi-evidence Searching (ZH)	0.808	0.808	0.866	0.866	0.887	0.889
Multi-evidence Searching (EN)	0.789	0.790	0.904	0.905	0.784	0.784
Multi-evidence Reasoning (ZH)	0.597	0.601	0.719	0.719	0.606	0.634
Multi-evidence Reasoning (EN)	0.757	0.769	0.373	0.404	0.457	0.468

Table 7: The F1@32 and F1@M performance on the Counting-Stars.

the results of Claude3 Opus, as shown in Figure 2. Hence, we hypothesize that the *lost-in-the-middle* phenomenon only occurs in specific tasks, length contexts, or models.

By observing the results of multiple experiments, we guess that the *lost-in-the-middle* phenomenon of LLMs is determined by their implicit reasoning or thinking patterns when dealing with specific tasks or length contexts. Interestingly, as illustrated in Figure 6 ("fail to follow instruction"), when collecting the numbers of stars, LLMs first attempt to memorize and recite relevant sentences and then further summarize them into the final result. According to the above findings, we guess this kind of implicit reasoning or thinking pattern may alleviate the *lost-in-the-middle* phenomenon.

5 Related Work

Prior research on long-context modeling has traditionally adopted perplexity as the primary evaluation metric (Peng et al., 2023; Chen et al., 2023). Meanwhile, synthetic tasks (e.g., retrieval tasks) have been employed to gauge the capacity of LLMs to handle extremely long inputs (Li et al., 2023a). However, as highlighted in Xiong et al. (2023), neither perplexity scores nor performance on synthetic tasks may fully capture the effectiveness of LLMs in real-world applications. Several benchmarks proposed by Bai et al. (2023b); An et al. (2023); Yuan et al. (2024); Qiu et al. (2024); Zhang et al. (2024) recently aim to evaluate long-context LLMs.

A recent benchmark for testing the long-context LLMs is needle-in-a-haystack, which asks LLMs to recite the information in a “needle” sentence (“The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day”) that is inserted at a designed location in a long text. The difference between the needle-in-a-haystack and existing benchmarks is that it does not rely on specific data, especially those that may be utilized to train LLMs. In addition, the needle-in-a-haystack can be treated as a benchmark where the test data can be easily replaced to mitigate the issue of data leakage, which generally occurs in existing long-context benchmarks. As mentioned before, however, many recently released LLMs evaluate the capability of long-context handling by testing the needle-in-a-haystack, all achieving nearly perfect performance, making it impossible to distinguish the gaps between different long-context LLMs.

The Counting-Stars evaluates the capabilities of multi-evidence searching and reasoning of LLMs, which should be more noteworthy in the long context modeling of LLMs, as reflected in tasks such as multi-document question answering and summarization. Concretely, the former primarily evaluates the capability of LLMs to collect multiple pieces of evidence simultaneously (distinguishing between long context and inserted sentences), while the latter tests the ability of LLMs to gather and reason various pieces of evidence at the same time correctly, that is, reasoning is required when collecting information (distinguishing between correct and incorrect evidence in inserted sentences). *To the best of our knowledge, the Counting-Stars is the first scalable long-context benchmark to ask LLMs to simultaneously differentiate between correct and incorrect evidence in each inserted sentence.*

Furthermore, similar to the recent benchmark (Li et al., 2023b), we refer to the long-dependency tasks as those that require capturing and understanding inter-dependency across multiple pieces of evidence spanning the entire long context. Hence, the Counting-Stars can also be considered a long-dependency task when calculating scores from the sample level, i.e., one testing sample only computes one score. In addition, since sentences are pieces of evidence and distributed throughout the entire long context, it is expected that other abilities behind long-context LLMs could be analyzed, including the long-context processing strategies and attention mechanisms, which is meaningful for studying the capability of long-context LLMs. It is worth mentioning that the cost of the Counting-Stars is lower than that of the needle-in-a-haystack, which is beneficial for reducing carbon emissions.

6 Conclusion

In this paper, we propose Counting-Stars, a multi-evidence, position-aware, and scalable benchmark for evaluating the multiple pieces of evidence retrieval capabilities of LLMs via two counting-based tasks. Utilizing the Counting-Stars, we conduct intriguing analyses on the behavior of LLMs, including the length-stability dilemma and the absence of the "lost-in-the-middle" phenomenon. Our analysis provides valuable insights into how LLMs handle long contexts, which can inform and guide future research endeavors.

7 Limitations

While this paper offers some insights into the performance of long-context LLMs, it may not be sufficiently diverse or extensive to provide a comprehensive evaluation of the long-context capabilities of LLMs, a constraint common to most analyses and benchmarks. However, we consider that for long-context scenarios, the capability to acquire multi-evidence is the most critical capability, which is also the central aspect tested by the Counting-Stars. Finally, we analyze and summarize potential uncertainty in our experiments, experiments on more famous LLMs.

(1) **Potential uncertainty in our experiments.**

(a) Context Used.

Through our experiments, it has been discovered that for tests like needle-in-a-haystack and Counting-Stars, different contexts may cause variations in the results. However, all experiments use the same context information in this paper. It must be noted, though, that different LLMs show variations in performance across different contexts.

(b) Prompt Used.

It is well known that LLMs are very sensitive to the design of prompts, and the results of different prompts can vary greatly. However, our experiments only constructed reasonable prompts that clearly express the task requirements without deliberately optimizing the prompts. From the experimental results, each model understood the prompts correctly without ambiguity.

(c) Service Used.

Due to regional access restrictions on the tested LLMs in this paper, two different services are used to test the five LLMs discussed in this paper: API and poe.com. Concretely, the latter approach does not allow adjusting model parameters, such as temperature. Therefore, when using the former, we set the temperature to 0 to ensure, as much as possible, the fairness of evaluation settings. However, using different access approaches may introduce some hard-to-find issues, potentially leading to biases in the testing results.

(d) Evaluation Used.

Actually, the adopted evaluation metric in this paper seems too simple, particularly in the multi-evidence reasoning task. A more comprehensive and reasonable evaluation metric may better reflect LLMs' long-context capability, such as using different context lengths as weights.

(2) **Experiments on more LLMs.**

In the future, we will evaluate more famous LLMs on the Counting-Stars, such as Llama3[8], Mistral[9], Llama2 (Touvron et al., 2023), Mixtral (Jiang et al., 2024), Command-R[10], LongLoRA (Chen et al., 2024), LongAlpaca(Chen et al., 2024), LWM (Liu et al., 2024), Qwen (Bai et al., 2023a), DeepSeek-V2 (DeepSeek-AI, 2024), etc.

(3) **Future expansion of the Counting-Stars.**

Initially, the Counting-Stars is designed to require LLMs to count the total number of stars in all sen-

[8] https://github.com/meta-llama/llama3

[9] https://mistral.ai/news/la-plateforme/

[10] https://docs.cohere.com/docs/command-r#model-details

tences inserted in the long context, which aims to test the multi-evidence searching of LLMs from a long dependency perspective. However, we find that if LLMs are required to calculate the total number of stars, they usually perform poorly. Specifically, we analyze the reasons for the bad performance, which mainly include three points:

- LLMs are unable to discover the sentences describing the little penguin counted stars.
- LLMs are able to discover all sentences but cannot remember them all.
- LLMs are able to remember all sentences but need better mathematical ability to calculate the total numbers correctly.

Still, we find that even if it is a simple mathematical problem of calculating "1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1", the probability of LLMs calculating correctly is lower. So, introducing the summation operation may be a simple and direct extension of the Counting-Stars. However, this paper currently focuses on the multi-evidence retrieval abilities of LLMs in long contexts. Therefore, we have chosen to require LLMs to list all the numbers of stars.

To further optimize and enhance the Counting-Stars, we imagine other evaluation strategies similar to the Counting-Stars, such as scattering stars by different players and specifying LLMs to search for the stars counted by one of them. Based on the above idea, adding more complex interactions between players may construct a more difficult question for evaluating LLMs.

Acknowledgments

We thank the three anonymous reviewers for carefully reading our paper and their insightful comments and suggestions.

References

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. *CoRR*, abs/2307.11088.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023a. Qwen technical report. *Preprint*, arXiv:2309.16609.

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. 2023b. Longbench: A bilingual, multitask benchmark for long context understanding. *CoRR*, abs/2308.14508.

Ramakrishna Bairi, Atharv Sonwane, Aditya Kanade, Vageesh D. C, Arun Iyer, Suresh Parthasarathy, Sriram K. Rajamani, Balasubramanyan Ashok, and Shashank Shet. 2023. Codeplan: Repository-level coding using llms and planning. *CoRR*, abs/2309.12499.

Avi Caciularu, Matthew E. Peters, Jacob Goldberger, Ido Dagan, and Arman Cohan. 2023. Peek across: Improving multi-document modeling via cross-document question-answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 1970–1989. Association for Computational Linguistics.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023. Longlora: Efficient fine-tuning of long-context large language models. *CoRR*, abs/2309.12307.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2024. Longlora: Efficient fine-tuning of long-context large language models. *Preprint*, arXiv:2309.12307.

DeepSeek-AI. 2024. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *Preprint*, arXiv:2405.04434.

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. 2024. Data engineering for scaling language models to 128k context. *CoRR*, abs/2402.10171.

Yunpeng Huang, Jingwei Xu, Zixu Jiang, Junyu Lai, Zenan Li, Yuan Yao, Taolue Chen, Lijuan Yang, Zhou Xin, and Xiaoxing Ma. 2023. Advancing transformer architecture in long-context large language models: A comprehensive survey. *CoRR*, abs/2311.12351.

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

Dacheng Li, Rulin Shao, Anze Xie, Ying Sheng, Lianmin Zheng, Joseph Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. 2023a. How long can context length of open-source LLMs truly promise? In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2023b. Loogle: Can long-context language models understand long contexts? *CoRR*, abs/2311.04939.

Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. 2024. World model on million-length video and language with blockwise ringattention. *Preprint*, arXiv:2402.08268.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts. *CoRR*, abs/2307.03172.

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

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

Zexuan Qiu, Jingjing Li, Shijue Huang, Wanjun Zhong, and Irwin King. 2024. Clongeval: A chinese benchmark for evaluating long-context large language models.

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

Mingyang Song, Haiyun Jiang, Lemao Liu, Shuming Shi, and Liping Jing. 2023a. Unsupervised keyphrase extraction by learning neural keyphrase set function. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 2482–2494. Association for Computational Linguistics.

Mingyang Song, Haiyun Jiang, Shuming Shi, Songfang Yao, Shilong Lu, Yi Feng, Huafeng Liu, and Liping Jing. 2023b. Is chatgpt A good keyphrase generator? A preliminary study. *CoRR*, abs/2303.13001.

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. *Preprint*, arXiv:2403.11802.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

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, Sinong Wang, and Hao Ma. 2023. Effective long-context scaling of foundation models. *CoRR*, abs/2309.16039.

Tao Yuan, Xuefei Ning, Dong Zhou, Zhijie Yang, Shiyao Li, Minghui Zhuang, Zheyue Tan, Zhuyu Yao, Dahua Lin, Boxun Li, Guohao Dai, Shengen Yan, and Yu Wang. 2024. Lv-eval: A balanced long-context benchmark with 5 length levels up to 256k. *CoRR*, abs/2402.05136.

Xingdi Yuan, Tong Wang, Rui Meng, Khushboo Thaker, Peter Brusilovsky, Daqing He, and Adam Trischler. 2020. One size does not fit all: Generating and evaluating variable number of keyphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7961–7975. Association for Computational Linguistics.

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. 2024. ∞bench: Extending long context evaluation beyond 100k tokens. *Preprint*, arXiv:2402.13718.