TAIL: A Toolkit for Automatic and Realistic Long-Context Large Language Model Evaluation

Gefei Gu[1] **Yilun Zhao**[1] **Ruoxi Ning**[2] **Yanan Zheng**[1] **Arman Cohan**[1,3]
[1]Yale University [2]University of Warterloo [3]Allen Institute for AI

Abstract

As long-context large language models (LLMs) gain increasing attention for their ability to handle extensive inputs, the demand for effective evaluation methods has become critical. Existing evaluation methods, however, fall short: needle-in-a-haystack (NIAH) and its variants are overly simplistic, while creating realistic benchmarks is prohibitively expensive due to extensive human annotation requirements. To bridge this gap, we propose TAIL, an automatic toolkit for creating realistic evaluation benchmarks and assessing the performance of long-context LLMs. With TAIL, users can customize the building of a long-context, document-grounded QA benchmark and obtain visualized performance metrics of evaluated models. TAIL has the advantage of requiring minimal human annotation and generating natural questions based on user-provided long-context documents. We apply TAIL to construct a benchmark encompassing multiple expert domains, such as finance, law, patent, and scientific literature. We then evaluate four state-of-the-art long-context LLMs using this benchmark. Results show that all the evaluated LLMs experience varying degrees of performance degradation as context lengths increase.

https://github.com/yale-nlp/TAIL

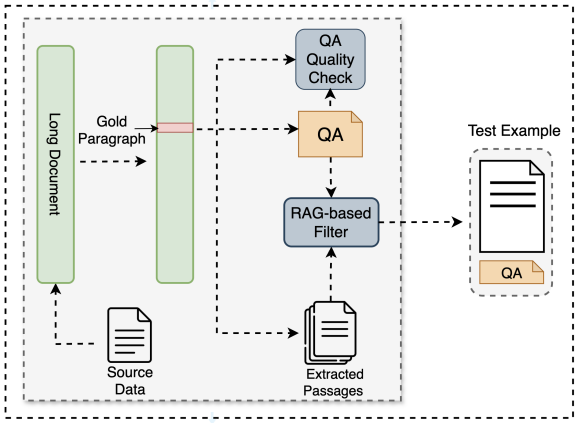

Figure 1: TAIL's workflow begins by constructing a long document from user-provided source data. It then identifies multiple "gold paragraphs" at various depths within this haystack. Using these gold paragraphs, TAIL generates high-quality question-answer pairs through its QA generation module, ensuring that each pair corresponds to a single identified paragraph. These pairs are then verified by the quality check module. The RAG-based filter further refines the collection by removing QA pairs that can be answered using multiple paragraphs within the test examples. Finally, TAIL assembles a benchmark from the remaining high-quality QA-test document pairs.

1 Introduction

The rise of long-context large language models (LLMs) has opened new possibilities for applications requiring comprehensive understanding and processing of extensive input context (Liu et al., 2023; Ding et al., 2023; Su et al., 2023; Peng et al., 2023; Gu and Dao, 2024). However, the evaluation of long-context LLMs poses unique challenges.

A line of research involves directly inserting specific document-irrelevant information into lengthy documents and querying about them, *i.e.,* needle-in-a-haystack (NIAH) and its variants (Song et al., 2024; Hsieh et al., 2024; Kuratov et al., 2024). However, the content and style of the inserted text differ significantly from the original document. These substantial distribution differences do not reflect real-world scenarios when dealing with long contexts and could influence the evaluation of the LLMs' long-context capabilities. Furthermore, the NIAH method is too simplistic that current models easily achieve nearly 100% in the test. Another line of research, such as LongBench (Bai et al., 2024) and LV-Eval (Yuan et al., 2024), follows the traditional evaluation protocols and directly extends the context lengths of test data. However, the documents included are typically limited to a maximum of 40K tokens and require expensive human anno-

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing
System Demonstrations, pages 198–208

tation, making it challenging to extend to longer contexts based on user-specific needs.

To address the aforementioned limitations and build upon the existing lines of research, we develop **TAIL**, a **T**oolkit for **A**utomatic and Real**I**stic **L**ong-context LLM evaluation creating reliable and high-quality evaluation benchmarks for long-context LLMs automatically. TAIL is designed to generate natural and reliable QAs at specific depths of long documents for creating high-quality evaluation benchmarks. The main contributions of our work are as follows:

- We develop a new toolkit, TAIL, for automatic benchmark building and evaluation for long-context LLMs. TAIL offers the advantage of generating benchmarks of any length from user-provided documents while producing more natural QA pairs without inserting new information.
- We collect source documents from a variety of specialized domains and build a long-context evaluation benchmark using TAIL.
- We use TAIL to evaluate four long-context LLMs on the generated benchmark. Our experimental results reveal that all the evaluated LLMs experience varying degrees of performance degradation as context lengths increase.

2 Related Work

Needle In A Haystack (NIAH) benchmarks, requiring models to retrieve randomly inserted sentences or facts within a long sequence, provide an automatic way to build a benchmark (Kamradt, 2023). Besides the vanilla NIAH task, advanced NIAH techniques are further developed, including techniques of multiple needles (Kuratov et al., 2024), confusing facts (Yuan et al., 2024), counting needles (Song et al., 2024) and simple reasoning (Hsieh et al., 2024). However, with automatically-generated QA pairs, current NIAH benchmarks suffer from the problem that questions are often irrelevant to the rest of the contexts and might be solved simply by retrieval instead of long-context reasoning and understanding abilities (Goldman et al., 2024).

Compared to NIAH benchmarks, realistic benchmarks comprise a wider range of tasks relevant to real-world needs by including tasks like summarization (Laban et al., 2024; Zhou et al., 2023), numerical reasoning (Zhao et al., 2024), and multi-hop reasoning (Wang et al., 2024; Ni et al., 2024) and spanning over multiple domains like code (Bogomolov et al., 2024), medical (Fan et al., 2024), novel (Wang et al., 2024; Karpinska et al., 2024), legal, finance, and etc. (Kwan et al., 2024). There also exist comprehensive realistic benchmarks encompassing multiple tasks like L-Eval (An et al., 2023), LongBench (Bai et al., 2024), LooGLE (Li et al., 2023), ∞Bench (Zhang et al., 2024) and BAMBOO (Dong et al., 2024). Due to the irregularity of question types, realistic benchmarks are usually either not long enough (less than 100k tokens), or expensive to collect and annotate. TAIL seeks to address both realistic needs and reduce human annotation when creating long-context benchmarks.

3 The TAIL Toolkit

This section provides an overview of the TAIL workflow[1], highlighting the main components and their interactions as illustrated in Figure 1. TAIL consists of three key components:

- **End-to-end Benchmark Generation.** This component consists of four steps. Given source data of varying lengths, TAIL first composes a long-context document (§3.1).

  TAIL then extracts multiple paragraphs at designated depths from the composed long documents as 'gold paragraphs' and generates QAs based on those paragraphs. Then, to ensure high-quality QA pairs, TAIL uses a quality checker to filter out QAs that cannot be correctly answered even if given the gold paragraph as references (§3.2).

  We now have a long document and QA pairs (together with their golden paragraphs containing the answer) at different locations in the long sequence. This long document has a maximum specified length. Finally, TAIL has an extraction module to further extract test data of different lengths from the long document, e.g., from 4K to maximum length set by users (§3.3). Note that these test data of different lengths coming from the same long-context documents share exactly the same QAs, thus ensuring control of variables when assessing long-context abilities.
- **Further Data Validation.** In addition to the low-quality QAs mentioned previously, since we want to test LLM's ability to generate answers towards a specific depth from the test document,

[1]We provide detailed documentation on using the TAIL at https://yale-nlp.github.io/TAIL/.

there are also other types of inappropriate QAs, such as QAs that can be answered by other paragraphs/chunks from test documents other than the gold paragraphs. To address this problem, TAIL utilizes another Retrieval Augmented Generation (RAG)-based filter to remove such inappropriate generations (§3.4).

- **Out-of-the-box LLM Evaluation and Performance Visualization.** While providing functionality for constructing long-context benchmarks, TAIL also implements an efficient pipeline for long-context evaluation and result visualization (*i.e.,* heatmap and line chart) (§3.5).

3.1 Long-context Document Preparation

TAIL is designed to compose long-context documents based on input texts of any length. The prepared input texts for constructing the long sequence are intended to meet the specified maximum length requirement of evaluated models.

For instance, if users want to generate a benchmark with 128k tokens to evaluate LLMs, input texts that are 128k tokens long are needed. If the texts users have prepared aren't long enough to meet the above requirement, we suggest combining multiple shorter inputs that are similar to each other in content. Users can select texts from the same domain or with related topics to create a cohesive, longer-context document. This approach ensures the combined text maintains coherence and relevance while providing sufficient length for the benchmark, and is similar to those in Kamradt (2023), where they build a long document using 218 essays from Paul Graham for the NIAH test.

3.2 QA generation

TAIL generates question-answer pairs using the long-context document provided by users through a three-step process. First, it extracts paragraphs from the long document according to the depth list (*i.e.,* locations in the input) the user specifies. Next, it creates QA pairs based on these selected paragraphs. Finally, TAIL checks the quality of the generated QA pairs, regenerating any that are deemed low-quality.

Gold Paragraph Extraction Rather than using original paragraphs to generate questions, we first divide the long document into equal-length segments and use these to generate QAs. We refer to these segments as "paragraphs" throughout the text. In practice, the segment size is set to 600 words to ensure each segments contains enough information to generate a relevant question. Secondly, based on the depth list provided by the users, TAIL extracts the chunks at these specified depths to serve as "gold paragraphs".

LLM-based QA Generation After obtaining all the gold paragraphs, we use GPT-4o to generate multiple-choice questions based on each individual gold paragraph. The specific prompt used for this stage is provided in Figure 2 in the appendix. In pilot study we found that GPT-4o is capable of generating reasonable QA pairs and we further perform a filtering step to only retain high quality questions. When generating QA pairs, we ensure that each question is based on only one gold paragraph. We set the number of choices for each question to six, with only one correct answer, which will reduce the chances of correct answers through random guessing. We chose multiple choice format as opposed to free form generation as it facilitates directly calculating performance metrics.

Quality Checking To filter out low-quality questions-answer pairs GPT-4o may generate, TAIL facilitates a quality check procedure. We prompt GPT-4o to answer each question based on the gold paragraph which used to generate this question. The specific prompt is provided in Figure 7 in the appendix. If a QA pair cannot be correctly answered in this step, it is considered potentially low-quality and the module iterates back to generate a new pair based on the same gold paragraph. This process may repeat several times, ultimately resulting in higher-quality QA pairs that accurately reflect the content of their respective gold paragraphs with no confusion. However, some gold paragraphs may be unsuitable for QA generation (*e.g.,* those containing minimal information), which could lead to an infinite loop. To prevent this, we've implemented a stopping mechanism that triggers after five unsuccessful attempts. In such cases, we replace the current gold paragraph with the preceding text chunk to serve as the new gold paragraph.

Human Validation To further examine the quality of the generated QA pairs, we randomly select 100 out of the total 400 generated QA pairs and assign human evaluators to examine their quality. The detailed validation procedure is listed in appendix A.2 . The results show that 92% of the samples are both clear and correct, indicating high

quality of the generated benchmark.

Question Generation Prompt

`[System Input]:`
You are a helpful AI bot that generates simple and detailed multiple choice question and its respective answer based on a context in the format of JSON. Each question should have options A, B, C, D, E and F and only one correct answer. Question should be clear and has no confusion. Answers should be a single character.

`[User Input]:`
Here's the context:
{context}
Please respond in the format of json.

Figure 2: Example of Prompt for QA generatetion in §3.2.

3.3 Test Example Construction

Our test example is defined as a question with a test document containing evidence for the question. After obtaining the long-context document and high-quality QAs (each with a gold paragraph), TAIL then utilizes them to construct test examples of different lengths. It's noteworthy that we generate test examples of different lengths using the same long-context document. This is necessary because when evaluating the long-context capabilities, it's important to test at various lengths while keeping other variables (*e.g.,* difficulty of the problem) constant. Such a strategy ensures consistent control over questions and documents. Given a question together with its gold paragraph, a predefined document length, and a question depth, this component automatically extracts related passages from the long document. These extracted passages meet the required depth and length conditions.

These extracted passages together with questions serve as test examples for the benchmark. LLMs are then evaluated to answer each question given the corresponding test documents. Since test documents are created through extraction, their maximum length is guaranteed not to exceed the length of the haystack.

To better illustrate how different components collaborate to generate QA pairs and test examples, the algorithms for QA generation and text example construction are provided in 1.

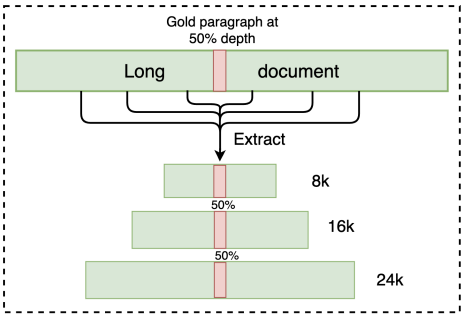

Figure 3: Illustrations demonstrating how the text example formulation module works to build test examples of varying lengths while maintaining the gold paragraph at a consistent depth from the long document.

3.4 RAG-based Filter

In some cases, a question might be answerable by multiple other paragraphs in the document, allowing LLMs to derive correct answers without the specific gold paragraph. For example, in a patent document, an author may highlight two key advantages of his invention at the beginning and elaborate on each benefit in subsequent paragraphs. Even if the beginning paragraph is omitted, LLMs could potentially obtain the correct answer by piecing together information from the remaining paragraphs. We need to avoid these questions as we aim to evaluate a model's ability to answer each question based on a paragraph from a specific depth. To ensure each question is answerable by only one specific paragraph from the test example, we implemented a RAG-based filter within TAIL. After obtaining QA, we use embedding models, *i.e.,* text-embedding-3-large (OpenAI, 2024b), to embed them and calculate cosine similarity to extract the top 5 related paragraphs from the test document (we make sure the paragraph that used to generate this QA is excluded). We ask GPT-4o to answer the QA based on these paragraphs. QA passes the test if GPT-4o cannot generate the correct answer given the top 5 related paragraphs, otherwise we will switch to QA generation module to regenerate another QA. Following the same strategy in §3.2, we set a stop mechanism to avoid infinite loop and replace the current gold paragraph with the preceding text chunk to serve as the new gold paragraph.

3.5 LLM Evaluation and Result Visualization

TAIL provides a ready-to-use evaluation module that enables users to easily test state-of-the-art LLMs on their generated benchmarks. We implement open-source models using vLLM (Kwon

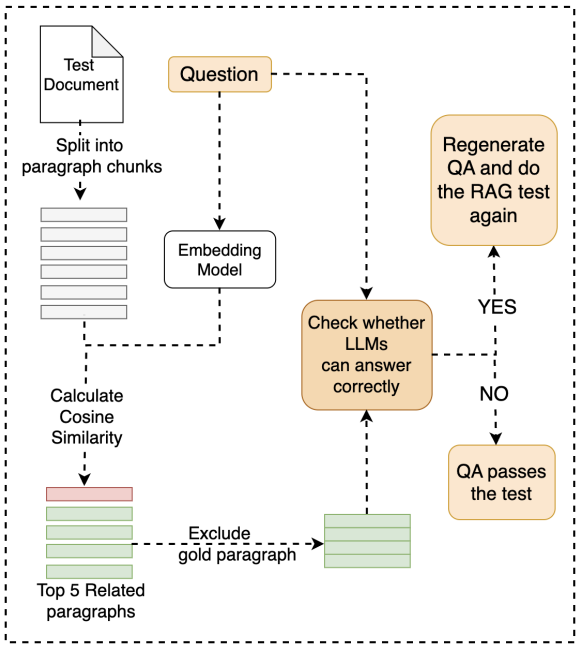

Figure 4: A demonstration on how the RAG-based filter works to filter out questions that can be answered by multiple paragraphs in the test example.

et al., 2023), and OpenAI API interface for commercial LLMs. For each benchmark question, models are prompted to think step by step and give answers given the long context input and the question. Then we use GPT-4o-mini to map the LLM-generated output to one of the multiple-choice options. To balance the mitigation of randomness, we set the temperature to 0 for inference.

TAIL provides several visualization tools, including heatmap graph, line chart, and weighted average scores. (1) Heatmap graph, similar to the visualization in NIAH, is a 2D box graph used to observe LLMs' performance in different depths and context lengths intuitively. (2) Line chart is used to compare LLMs' performance across context lengths. (3) Following the model ranking criteria introduced by RULER (Hsieh et al., 2024), TAIL offers two weighted average scores to aggregate model performance across various context sizes: wAvg. (inc) and wAvg. (dec). In wAvg. (inc), the weight linearly increases with sequence length, while in wAvg. (dec), it linearly decreases. The wAvg. (inc) score emphasizes models' ability to handle longer texts, whereas the wAvg. (dec) score focuses more on their performance with shorter texts. This dual scoring approach provides a comprehensive evaluation of model performance across various text lengths. We provide the algorithm for TAIL workflow in Algorithm 1.

Algorithm 1 QA Generation and Quality Checking

Require: Input: D, T, L
{D denotes the user-defined target depth set, T denotes the target token length set, and L denotes the long document.}
Ensure: Output: QA and documents pairs

for depth **in** D **do**
  gold_paragraph ← find_paragraphs(L, depth)
  QA ← generate_QA(gold_paragraph)
  if GPT-4o cannot answer QA correcly based sorely on the gold_paragraph **then**
    regenerate a new QA
  end if
  rag context ← top 5 related paragraphs to QA(exclude gold paragraph)
  if GPT-4o can correctly answer the question based on rag context **then**
    regenerate a new QA
  end if
  for token_length **in** T **do**
    test document ← extract_passage(depth, token_length)
    return { QA, test document}
  end for
end for

Domain	Source Document Numbers	Average Token Lengths per Doc	Question Numbers
Finance	10	90.5k	190
Patent	10	74.7k	190
Legal	10	68.2k	190
Paper	30	18.7k	190

Table 1: Statistic of the TAIL-constructed benchmark.

4 Experiments and Results

Next, we demonstrate how TAIL is utilized to evaluate nine long-context LLMs across four specialized domains: finance, patents, legal, and scientific papers. We present the results for these nine evaluated LLMs and provide a detailed analysis.

4.1 Benchmark Construction

To make our benchmark fit into real-world scenarios, we collected a variety of source documents from four expert domains, including government financial reports, patent documents, legal documents from Scotland Court, and scientific papers from Arxiv. We retained plain text while removing figures and tables. This decision was made for two reasons: firstly, some models are not multimodal and cannot process images; secondly, tables may require specialized reasoning ability but we only want to test LLMs ability to process plain texts. All the collected documents are released in 2024 to mitigate pre-training data contamination for the models being evaluated. We used TAIL to generate documents ranging from 8k to 128k tokens, increasing in 8k-token increments for each domain. The

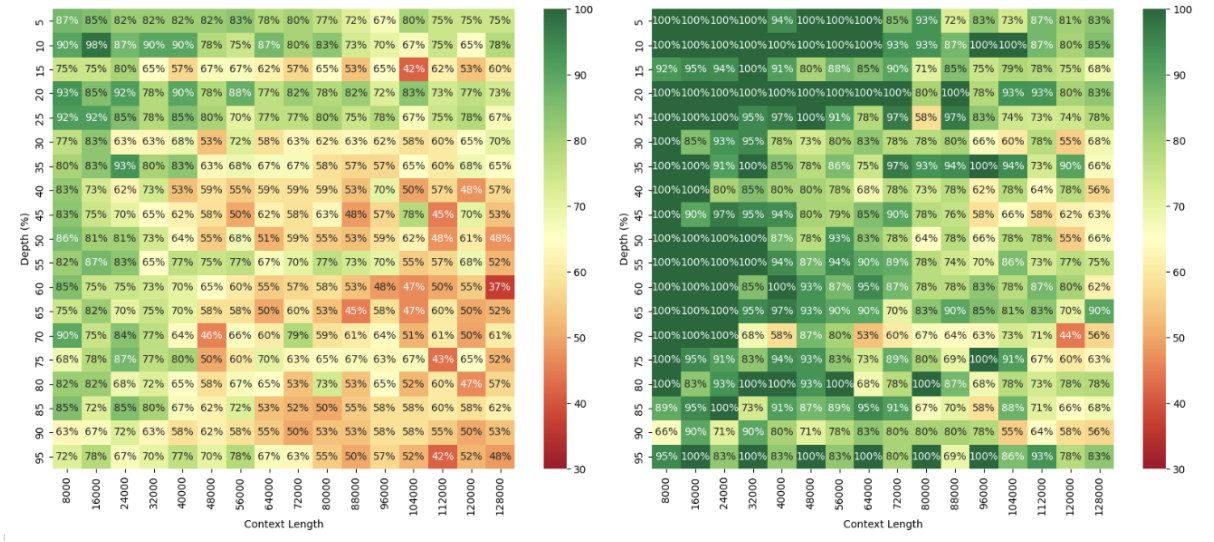

Figure 5: Heatmap showing the average results of two LLMs on the cross-domain benchmark generated with TAIL. The left panel shows results for GLM-4-9B-chat, while the right panel displays results for GPT-4o.

maximum length was set to 128k tokens, aligning with the context limits of most LLMs being evaluated at the time of writing. We generated questions at varying depths throughout each document, starting at 5% and increasing in 5% increments up to 95%. The detailed statistic of our benchmark is shown in Table 1. [2]

4.2 Models & Inference Setup

We evaluated two commercial and seven open-source long-context LLMs on the constructed benchmark: GPT-4o (OpenAI, 2024a), Gemini-1.5-flash (Gemini, 2024), LLaMA-3.1-8B-Instruct, LLaMA-3.1-70B-Instuct-AWQ(AI@Meta, 2024), GLM-4-9B-chat (GLM et al., 2024), Qwen2-7B-Instruct(Yang et al., 2024), Qwen2.5-72B-Instruct-AWQ(Team, 2024), Phi-3-small-128k (Abdin et al., 2024) and Llama-3-8B-ProLong-512k-Instruct(Gao et al., 2024). All the evaluated models support context lengths of up to 128k tokens, with the exception of Llama-3-8B-ProLong-512k-Instruct and Gemini-1.5-Flash, which support up to 512k and 1 million tokens, respectively. TAIL evaluates all open-source models using vLLM (Kwon et al., 2023), while utilizing API calls for commercial LLMs. For our inference process, we set the temperature parameter to 0 and limit the maximum output to 512 tokens. The prompt for testing is provided in Figure 7 in the appendix.

[2]We realized the TAIL generated benchmark on huggingface at https://huggingface.co/datasets/yale-nlp/TAIL.

4.3 Results

The main results are in Table 2, which shows the long-context performance of different LLMs at various context lengths. Figure 6 demonstrate each models' performance across different depths and context lengths. Figure 5 presents heatmaps illustrating long-context scores of different depths and lengths. Our main findings are as follows.

All LLMs experience performance degradation as the context lengths increase on the benchmark. The top-performing model on this benchmark is GPT-4o, with an average accuaracy of 88.84%. Qwen2.5-72B-Instruct, which leverages YaRN to enhance model length extrapolation and has a large parameter size, stand out to be the best performing open-source model we tested. Though the top 4 models we tested can achieve over 90% accuracy when processing 8k tokens length document, their accuracy drops to less then 70% when the document context length extends to 128k tokens. For other open-source models with fewer than 10 billion parameters, accuracy drops to around 60% when context lengths exceed 64k tokens.

The Benchmark generated by TAIL is more challenging than NIAH To demonstrate our advantages over the standard NIAH test, we use the same input document to build two benchmarks using both our method and the NIAH method. We evaluate GPT-4o on these two benchmarks, as illustrated in Figure 11 in the appendix, while GPT-4o

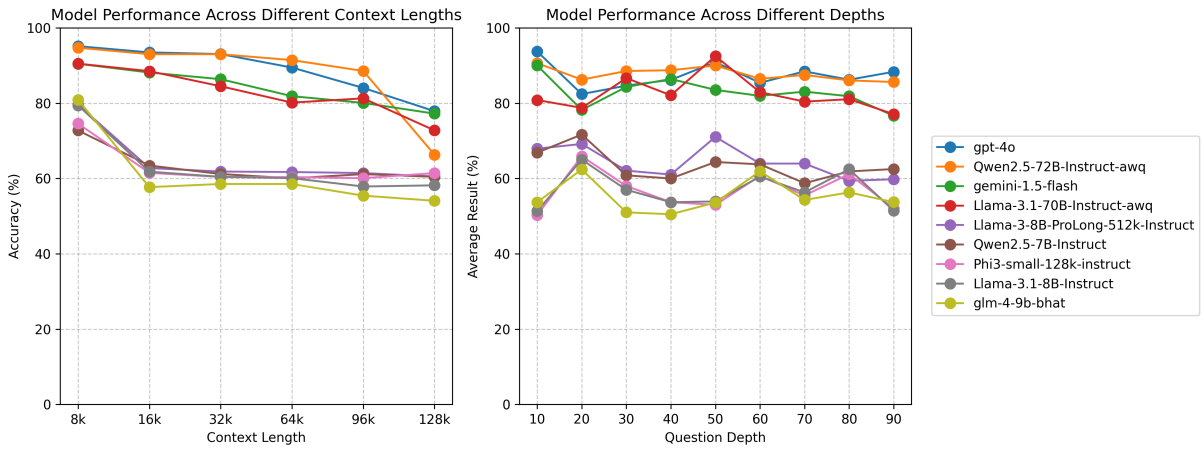

Figure 6: Analysis of accuracy across different context lengths and different question depths.

Models	8k	16k	32k	64k	96k	128k	Avg	wAvg (inc)	wAvg (dec)	128k/8k (%)
GPT-4o	**95.14**	**93.48**	**93.04**	89.46	84.05	**77.87**	**88.84**	**84.29**	**92.84**	81.85
Qwen2.5-72B-Instruct-awq	94.73	93.00	93.00	**91.44**	**88.55**	66.32	87.87	81.58	92.67	70.01
Gemini-1.5-flash	90.47	88.14	86.37	81.86	80.07	77.22	84.02	80.54	87.47	**85.35**
Llama-3.1-70B-Instruct-awq	90.50	88.50	84.50	80.15	81.28	72.77	82.95	78.75	87.02	80.41
Llama-3-8B-ProLong-512k-Instruct	79.61	62.76	61.84	61.71	61.44	60.52	64.65	61.67	68.65	76.02
Qwen2.5-7B-Instruct	72.76	63.44	61.18	60.05	61.15	60.44	63.17	61.07	65.99	83.07
Phi-3-small-Instruct	74.58	61.49	60.40	60.37	60.08	61.43	63.06	61.07	65.99	82.37
Llama-3.1-8B-Instruct	79.43	61.81	60.44	60.08	57.87	58.18	62.97	59.32	67.68	73.25
GLM-4-9B-Chat	80.91	57.70	58.54	58.52	55.44	54.07	60.86	56.49	66.38	66.83

Table 2: Performance of different models at various context lengths, sorted by Average Acc in descending order. The Average Acc column shows the average accuracy across all context lengths, and the last column shows the ratio of average accuracy on 128k-token documents to 8k-token documents. Bold numbers indicate the highest value in each column, while underlined numbers indicate the second highest.

achieves nearly 100% accuracy performance in the NIAH test, our benchmark reveals how its performance declines when dealing with long-context documents. GPT-4o remains over 93% accuracy when the context length is less than 32k tokens, but when context lengths extends to 128k, it cannot achieve more than 80% accuracy.

LLMs vary in their ability to maintain performance as context length increases. We present the ratio of each model's performance on 128k-token documents compared to 8k-token documents in Table 2. Gemini-1.5-Flash stands out for its strong ability to maintain performance, retaining an impressive 85.35% of its 8k tokens performance at 128k tokens. In contrast, Qwen2.5-72B-Instruct-awq achieves high performance when dealing documents that less then 96k tokens, but has a signigicant performance drop when contexts reachs 128k tokens. Additionally, as seen in Figure 6 , weaker models often exhibit an early performance drop. For example, glm-4-9b-chat's performance declines 27.8% when the context length extends from 8k to 16k, whereas stronger models tend to experience a later drop or show no significant drop.

5 Conclusion

The emergence of long-context LLMs has highlighted the need for more effective evaluation tools. In this paper, we propose TAIL, an automatic and realistic toolkit for long-context large language model evaluation. TAIL can generate benchmarks end-to-end with the given source documents. Moreover, TAIL offers evaluation modules for testing and results visualization. We demonstrate TAIL's capabilities by creating a cross-domain benchmark, illustrating its effectiveness in both benchmark development and LLM performance evaluation. We believe that the TAIL will serve as a useful toolkit for evaluating long-context LLMs.

Acknowledgements

We are also grateful for the compute support provided by Microsoft Research's Accelerate Foundation Models Research (AFMR) program.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Amit Garg, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Chen Liang, Weishung Liu, Eric Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Xia Song, Masahiro Tanaka, Xin Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Michael Wyatt, Can Xu, Jiahang Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. Preprint, arXiv:2404.14219.

AI@Meta. 2024. The llama 3 herd of models. Preprint, arXiv:2407.21783.

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. Preprint, arXiv:2307.11088.

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. Preprint, arXiv:2308.14508.

Egor Bogomolov, Aleksandra Eliseeva, Timur Galimzyanov, Evgeniy Glukhov, Anton Shapkin, Maria Tigina, Yaroslav Golubev, Alexander Kovrigin, Arie van Deursen, Maliheh Izadi, and Timofey Bryksin. 2024. Long code arena: a set of benchmarks for long-context code models. Preprint, arXiv:2406.11612.

Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, Nanning Zheng, and Furu Wei. 2023. Longnet: Scaling transformers to 1,000,000,000 tokens. arXiv preprint arXiv:2307.02486.

Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. 2024. Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. Preprint, arXiv:2309.13345.

Yongqi Fan, Hongli Sun, Kui Xue, Xiaofan Zhang, Shaoting Zhang, and Tong Ruan. 2024. Medodyssey: A medical domain benchmark for long context evaluation up to 200k tokens. Preprint, arXiv:2406.15019.

Tianyu Gao, Alexander Wettig, Howard Yen, and Danqi Chen. 2024. Enabling large language models to generate text with citations.

Gemini. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. Preprint, arXiv:2403.05530.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. Preprint, arXiv:2406.12793.

Omer Goldman, Alon Jacovi, Aviv Slobodkin, Aviya Maimon, Ido Dagan, and Reut Tsarfaty. 2024. Is it really long context if all you need is retrieval? towards genuinely difficult long context nlp. Preprint, arXiv:2407.00402.

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? Preprint, arXiv:2404.06654.

Gregory Kamradt. 2023. Needle in a haystack - pressure testing llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack/tree/main.

Marzena Karpinska, Katherine Thai, Kyle Lo, Tanya Goyal, and Mohit Iyyer. 2024. One thousand and one pairs: A "novel" challenge for long-context language models. Preprint, arXiv:2406.16264.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. 2024. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. Preprint, arXiv:2406.10149.

Wai-Chung Kwan, Xingshan Zeng, Yufei Wang, Yusen Sun, Liangyou Li, Lifeng Shang, Qun Liu, and Kam-Fai Wong. 2024. M4le: A multi-ability multi-range multi-task multi-domain long-context evaluation benchmark for large language models. Preprint, arXiv:2310.19240.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In Proceedings of the 29th Symposium on Operating Systems Principles, pages 611–626.

Philippe Laban, Alexander R. Fabbri, Caiming Xiong, and Chien-Sheng Wu. 2024. Summary of a haystack: A challenge to long-context llms and rag systems. Preprint, arXiv:2407.01370.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. 2023. Loogle: Can long-context language models understand long contexts? Preprint, arXiv:2311.04939.

Hao Liu, Matei Zaharia, and Pieter Abbeel. 2023. Ring attention with blockwise transformers for near-infinite context. Preprint, arXiv:2310.01889.

Xuanfan Ni, Hengyi Cai, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, and Piji Li. 2024. Xl^{2}bench: A benchmark for extremely long context understanding with long-range dependencies. Preprint, arXiv:2404.05446.

OpenAI. 2024a. Hello gpt-4o.

OpenAI. 2024b. text-embedding-3-large. https://openai.com.

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, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Bolun Wang, Johan S. Wind, Stanislaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, Peng Zhou, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. 2023. Rwkv: Reinventing rnns for the transformer era. Preprint, arXiv:2305.13048.

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.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. Roformer: Enhanced transformer with rotary position embedding. Preprint, arXiv:2104.09864.

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

Cunxiang Wang, Ruoxi Ning, Boqi Pan, Tonghui Wu, Qipeng Guo, Cheng Deng, Guangsheng Bao, Xiangkun Hu, Zheng Zhang, Qian Wang, and Yue Zhang. 2024. Novelqa: Benchmarking question answering on documents exceeding 200k tokens. Preprint, arXiv:2403.12766.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. arXiv preprint arXiv:2407.10671.

Tao Yuan, Xuefei Ning, Dong Zhou, Zhijie Yang, Shiyao Li, Minghui Zhuang, Zheyue Tan, Zhuyu Yao, Dahua Lin, Boxun Li, et al. 2024. LV-Eval: A balanced long-context benchmark with 5 length levels up to 256k. arXiv preprint arXiv:2402.05136.

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.

Yilun Zhao, Yitao Long, Hongjun Liu, Ryo Kamoi, Linyong Nan, Lyuhao Chen, Yixin Liu, Xiangru Tang, Rui Zhang, and Arman Cohan. 2024. DocMath-eval: Evaluating math reasoning capabilities of LLMs in understanding long and specialized documents. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 16103–16120, Bangkok, Thailand. Association for Computational Linguistics.

Yijie Zhou, Kejian Shi, Wencai Zhang, Yixin Liu, Yilun Zhao, and Arman Cohan. 2023. Odsum: New benchmarks for open domain multi-document summarization. Preprint, arXiv:2309.08960.

A Appendix

A.1 Examples of Prompts Used

Model Evaluation Prompt

`[User Input]:`
I will give you a multiple choice question and a corresponding document. Please provide your thoughts step by step.
Questions: {question} Options: {options}

Please answer the above question refer to this document only:
{document}

Figure 7: Example of prompt for answering the questions in the developed benchmark (§3.5)

A.2 Human Validation Procedure

We randomly selected 100 samples for human evaluation to assess the correctness and clarity of each question in relation to its corresponding golden paragraph. Note that evaluators check the quality of questions based only on the golden paragraph, not the entire document. Evaluators were asked to examine samples based on the following criteria:

1. **Clarity within Context:** Does the question remain unambiguous when the gold paragraph is placed within a longer document? For example, questions using pronouns like "he" or "she" without clear antecedents were flagged as potentially ambiguous.

2. **Paragraph Suitability:** Is the gold paragraph suitable for generating a clear and reasonable question?

3. **Answerability:** Can the question be accurately answered using only the information provided in the gold paragraph?

4. **Specificity:** Does the question target information unique to the gold paragraph, rather than general knowledge or information?

5. **Linguistic Quality:** Is the question well-formed, grammatically correct, and free of spelling errors?

A question is considered high quality when it meets all of these criteria.

A.3 Visualization Results

As we discussed before, we tested nine LLMs on the benchmark we created. We provide heatmaps of the first two models (GLM-9B-128k-chat, GPT-4o) in Figure 5, and heatmaps for some of the other models (Qwen2.5-7B-Instruct, Llama3.1-8B-Instruct and Llama3.1-70B-Instruct-awq) are presented below:

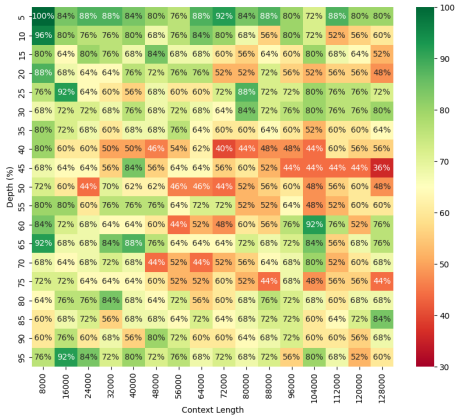

Figure 8: Heatmap showing Qwen2.5-7B-Instruct on the cross-domain benchmark generated with TAIL.

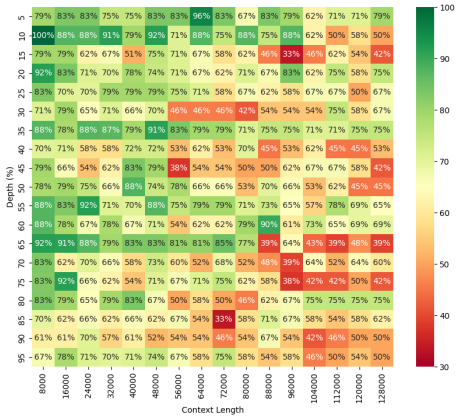

Figure 9: Heatmap showing Llama3.1-8B-Instruct on the cross-domain benchmark generated with TAIL.

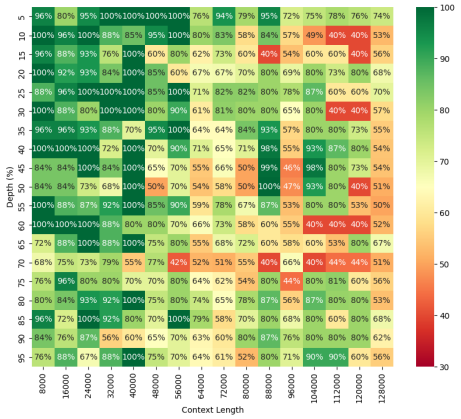

Figure 10: Heatmap showing Llama3.1-70B-Instruct-awq on the cross-domain benchmark generated with TAIL.

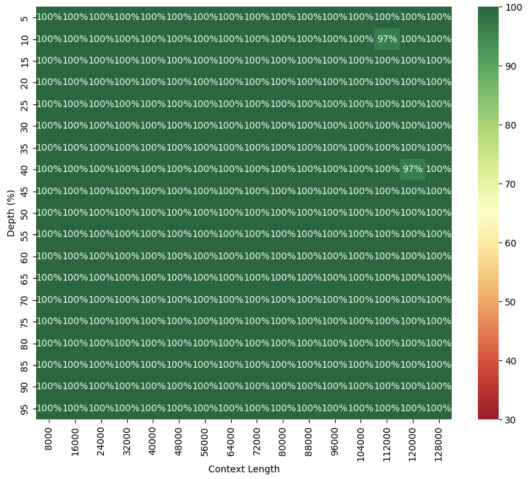

Figure 11: Heatmap showing GPT-4o's performance using NIAH method. Although GPT-4o achieves perfect performance on the standard NIAH test, it struggles with the TAIL-constructed benchmark, highlighting the challenges posed by our methods.