

LONGAGENT: Achieving Question Answering for 128k-Token-Long Documents through Multi-Agent Collaboration

Anonymous ACL submission

Abstract

Large language models (LLMs) have achieved tremendous success in understanding language and processing text. However, question-answering (QA) on lengthy documents faces challenges of resource constraints and a high propensity for errors, even for the most advanced models such as GPT-4 and Claude2. In this paper, we introduce LONGAGENT, a multi-agent collaboration method that enables efficient and effective QA over 128k-token-long documents. LONGAGENT adopts a *divide-and-conquer* strategy, breaking down lengthy documents into shorter, more manageable text chunks. A leader agent comprehends the user’s query and organizes the member agents to read their assigned chunks, reasoning a final answer through multiple rounds of discussion. Due to members’ hallucinations, it’s difficult to guarantee that every response provided by each member is accurate. To address this, we develop an *inter-member communication* mechanism that facilitates information sharing, allowing for the detection and mitigation of hallucinatory responses. Experimental results show that a LLaMA-2 7B driven by LONGAGENT can effectively support QA over 128k-token documents, achieving 16.42% and 1.63% accuracy gains over GPT-4 on single-hop and multi-hop QA settings, respectively.

1 Introduction

Nowadays, the capabilities of large language models (LLMs) have been rapidly advancing, driven by ever-increasing model sizes and data volumes (OpenAI, 2023). However, the prohibitively high training costs preclude these capability gains from extending to LLMs’ understanding of lengthy texts (Chen et al., 2023c). This poses a significant limitation on the practical applications of LLMs, such as querying information from books, analyzing legal documents or academic papers.

Researchers have primarily focused on two directions to address this issue. The first direction

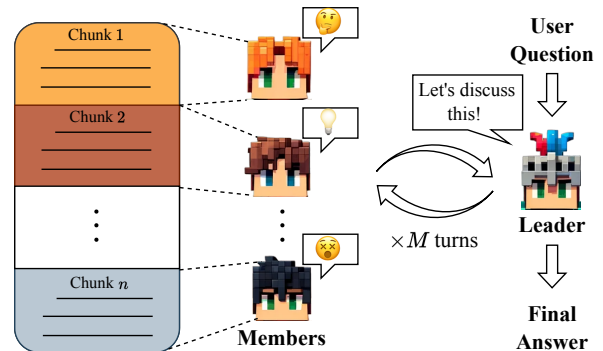


Figure 1: LONGAGENT collaboration scheme. The input long text (left) is segmented into chunks and assigned to corresponding members. The Leader receives user question (right), breaks them down into the simplest sub-question, convenes members for discussion, ultimately obtaining answers to all sub-question, and reasons to make the final response.

is to consider positional encoding as a crucial aspect (Press et al., 2022; Chen et al., 2023b; Peng et al., 2023; Chen et al., 2023a), taking techniques like extrapolation or interpolation to enable the positional encoding to handle *unseen* positions during pre-training. The second category employs more complex mechanisms like recurrent structures (Zhou et al., 2023; Zhang et al., 2024), token selection (Mohtashami and Jaggi, 2023; Tworowski et al., 2023), or sliding windows (Xiao et al., 2023; Han et al., 2023) to enable the limited context window to process longer input texts. Despite these efforts, effectively understanding lengthy texts and accurately answering user queries remains a challenging problem (Hsieh et al., 2024). Models tailored for long text processing may compromise their innate capabilities for understanding shorter texts (Jin et al., 2024) and tend to overlook critical information situated in the middle of long documents, a phenomenon known as *lost in the middle* (Liu et al., 2023).

In this paper, we propose LONGAGENT, a novel

065 approach for long-document QA. LONGAGENT
066 employs a *divide-and-conquer* strategy, breaking
067 down long documents into more manageable
068 smaller text chunks. Leveraging the charac-
069 teristic of aligned LLMs to follow instructions,
070 LONGAGENT adopts a multi-agent collaboration
071 approach to effectively support QA on ultra-long
072 documents (over 100k tokens). As shown in
073 Figure 1, the agent team consists of a leader
074 agent and multiple member agents. The leader
075 is responsible for understanding the user’s question
076 and directing the members to gather clues from
077 their assigned text chunks. Depending on the
078 complexity of the question, this process may
079 involve one or more rounds of interaction. The
080 interaction ends when the leader deems the clues
081 sufficient to reason the final answer. Managing
082 a large number of members is non-trivial due to
083 the model hallucination. We propose an *inter-*
084 *member communication* mechanism to identify the
085 members in a hallucinatory state and mitigate their
086 influence to the leader’s decision-making.

087 To comprehensively evaluate LONGAGENT, we
088 propose *Needle-in-a-Haystack PLUS*.¹ Compared
089 with original Needle-in-a-Haystack, it can test the
090 model’s capability of handling multi-hop QA tasks.
091 We have also tested LONGAGENT with all long-
092 document QA tasks of mainstream LongBench
093 (Bai et al., 2023) and InfiniteBench (Zhang et al.,
094 2023). The experimental results have demonstrated
095 the effectiveness of LONGAGENT.

096 The contributions of this paper are as follows:
097 1) we propose LONGAGENT, which enables 4k
098 context-driven LLMs to achieve QA on 128k
099 long documents; 2) we develop the *Needle-in-a-*
100 *Haystack Plus*, which enables comprehensive eval-
101 uation of LLMs’ capabilities in long-document QA.
102 3) our experimental results show that the LLaMA2-
103 7B model driven by LONGAGENT exhibits superior
104 long-text QA capabilities comparable to GPT-4.

105 2 LONGAGENT for Long-Document QA

106 2.1 Method Overview

107 We first formulate the problem that LONGAGENT
108 aims to solve. Given a document $d = \{w_i\}_{i=1}^n$
109 and a user’s question q , our goal is to build a QA
110 model that can respond to the question according
111 to the content mentioned in the document. The
112 challenge here is that the length n of document
113 d may be very long, even exceeding 100k tokens.

¹URL of the benchmark is omitted here pending the review.

114 Therefore, directly processing the entire document
115 d may incur high computational cost and inaccurate
116 response results.

117 LONGAGENT adopts a *divide-and-conquer* strat-
118 egy, breaking down the document d into $m =$
119 $\lceil n/l \rceil$ manageable short text chunks and assigning
120 them **one-to-one** to m member agents. The $\lceil \cdot \rceil$
121 represents the ceiling operator, while l represents
122 the predetermined chunk size (e.g., 1024 or 2048
123 tokens). l is set to be significantly smaller
124 than the context window size of the member
125 agents (4096 tokens for LLaMA2), thus reserving
126 ample room for subsequent multi-turn interactions.
127 Subsequently, leveraging the characteristic of
128 LLMs to follow instructions, a leader agent
129 and multiple member agents form a team that
130 collaboratively searches these text chunks for clues
131 and reasons to find answers. As illustrated in Figure
132 2, the collaboration consists of three steps, which
133 we elaborate on in detail in the subsequent sections.

134 2.2 Collaborative Reasoning

135 To answer a user’s question q , in each interaction
136 round, the leader generates an instruction and
137 broadcasts it to all members. The members
138 read their assigned document chunks and return
139 instruction-relevant cues. If the user’s question
140 is a complex multi-hop query, the above leader-
141 member interaction may proceed over multiple
142 rounds. Formally, given an user’s question q and
143 the interaction history $S_{i-1} = \{s_1, s_2, \dots, s_{i-1}\}$
144 from the previous $i - 1$ rounds, the i^{th} round of
145 interaction commences with a decision-making
146 process by the leader:

$$147 \quad a_i \sim \text{Leader}(a|S_{i-1}, q), \quad (1)$$

148 where $a_i \in \{\text{QUERY}, \text{CONFLICT}, \text{ANSWER}\}$
149 represents the decision outcome of the leader
150 agent in i^{th} round. If $a_i = \text{QUERY}$ ², it indicates
151 that the interaction history up to the $(i - 1)^{\text{th}}$
152 round is insufficient to answer the original question
153 q . The leader then initiates the next round of
154 interaction and provides a new instruction. As
155 illustrated in Figure 2, to answer *Which team*
156 *does the player named 2015 Diamond Head*
157 *Classic’s MVP play for?*, the leader first instructs
158 the members to identify who won the 2015
159 Diamond Head Classic MVP. Then, based on the
160 cues returned by members (i.e. Buddy Hield),

²CONFLICT and ANSWER actions are described in detail
in section 2.3 and 2.4 respectively.

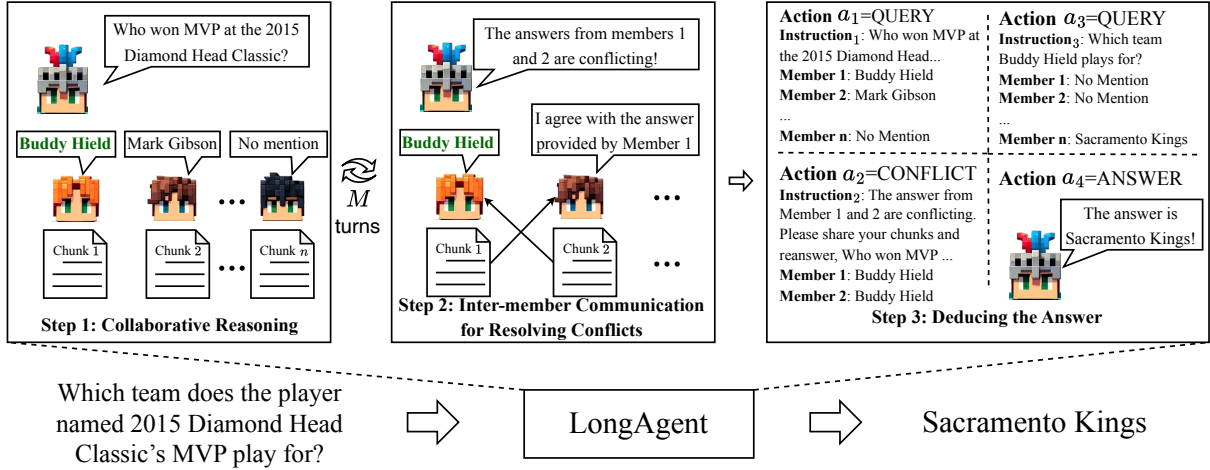


Figure 2: An Overview of LONGAGENT. In Step 1 and Step 2, the leader organizes the team to gather clues from the text chunks and resolve conflicts. After multiple rounds of iteration, the leader reasons out the final answer based on the information accumulated in the interaction history (corresponding to Action a_4 in Step 3).

it dynamically generates the next instruction to find *Which team Buddy Hield plays for*. This decomposition of complex queries and multi-round interaction is crucial, as relevant information may be scattered across different chunks of the lengthy document, precluding any single agent from directly answering the original multi-hop question in one interaction round.

2.3 Resolving Conflicts

Due to model hallucinations, members may respond with content not mentioned in their chunks. The interaction in Step 1 of Figure 2 serves as an example, where two members respectively believe *Buddy Hield* and *Mark Gibson* to be the MVP of the 2015 Diamond Head Classic, despite the latter not being mentioned in the text chunk. In such cases, the leader selects the `CONFLICT` action, which invokes the *inter-member interaction* mechanism to resolve the conflict. This mechanism is inspired by the following empirical findings: (a) When a chunk lacks clues relevant to the answer, the member tend to hallucinate responses instead of honestly admitting *No Mention*. (b) However, when the answer is present in the chunk, the member make mistakes less frequently. Therefore, we share text chunks from two members with conflicting responses, expecting the hallucinating agent to revise its response upon receiving the chunk mentioning the correct answer. Formally:

$$\text{Hallucination} = m_i(c_i), \text{ Truth} = m_j(c_j), \quad (2)$$

$$\text{Truth} = m_j(c_j \oplus c_i) = m_i(c_j \oplus c_i), \quad (3)$$

c_i and c_j respectively represent two text chunks, where c_j contains the correct answer while c_i does not. m_i and m_j denote two members. \oplus denotes concatenation of two chunks. Our experimental results demonstrate that sharing text chunks is a simple yet effective strategy. The majority of members experiencing hallucination tend to correct their original responses upon receiving the chunk containing the correct answers, resulting in accurate output.

Once the conflicts are resolved, the leader executes the decision process described in Section 2.2 again and selects the next action for the subsequent interaction round.

2.4 Deducing the Answer

When the leader judges that the interaction history contains sufficient information to reason the answer, it executes the action $a = \text{ANSWER}$ and generates the final answer. Taking the interaction history from Figure 2 (Step 3) as an example, in the first round, the leader performs $a_1 = \text{QUERY}$ and learns that both *Buddy Hield* and *Mark Gibson* are potential candidates for the 2015 Diamond Head Classic MVP. Consequently, in the second round, it executes $a_2 = \text{CONFLICT}$ and discovers that *Mark Gibson* is a hallucinated answer. Subsequently, in the third round, the leader performs $a_3 = \text{QUERY}$ and finds that *Buddy Hield* plays for the *Sacramento Kings*. At this point, the interaction history contains enough information to derive the answer to the original question, so the leader executes $a_4 = \text{ANSWER}$ and

outputs *The final answer is Sacramento Kings.*

2.5 How to Construct Agents

LONGAGENT involves both leader agents and member agents, which can be obtained by fine-tuning open-source base models or prompting strong instruction-following models.



Fine-tuning open-source base models. We fine-tune the LLaMA2-7B model to build our agent team. To train the leader agent, we generated 1,000 interaction trajectories using GPT-4 and manually verified the correctness of these trajectories. Based on this data, the leader agent learns how to decompose complex problems and reason the final answer from interaction history. To train the members to read their assigned text chunks and respond to the leader’s instructions, we constructed a 25,000-sample QA dataset based on the SQuAD dataset (Rajpurkar et al., 2016). Each sample comprises a text chunk and a corresponding question. Among these samples, 10,000 text chunks contain the answer to the associated question, while the remaining 15,000 do not mention the answer. For the latter cases, the member is trained to respond with *no mention*.

Prompting instruction-following models. For aligned models (such as GPT-4, GPT-3.5, etc.), we can use prompting to construct an agent team. For example, a viable prompt for the member agent is *You are an QA expert, adept at searching for relevant information from given documents and providing answers.* Please refer to Appendix B for full prompt templates and interaction trajectories.

3 Experimental Setup

3.1 Evaluation Protocol

Needle-in-a-Haystack PLUS: The *Needle-in-a-Haystack* (Kamradt, 2023) is currently one of the most popular testbed for evaluating LLM’s capability to handle long documents. As illustrated in Figure 3, a text (the *needle*) is inserted into a lengthy document (the *haystack*). The *Needle-in-a-Haystack* evaluates a model’s ability to retrieve critical information by testing whether it can accurately answer a given question. In its setup, the **length of haystack** L is defined as the document’s token count, while the needle’s insertion position is determined by the **depth percentage** α , which represents the percentage of tokens preceding the insertion point relative to the total token count L . By varying L and α , *Needle-in-a-Haystack*

: The Haystack : The Needle

... The first step is to decide what to work on. The work you choose needs to have three qualities: it has to be something you have a natural aptitude for, that you have a deep interest in, and that offers scope to do great work. **The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day.** In practice you don't have to worry much about the third criterion. Ambitious people are if anything already too conservative about it. So all you need to do is find something you have an aptitude for and great interest in. ...

Question: What is the best thing to do in San Francisco?

Figure 3: Overview of the *Needle in a Haystack*. By varying the depth percentage α and the haystack length L , we can conveniently construct test samples of different lengths, with critical information situated at varying positions.

: The Haystack : The Needle1 : The Needle2

... The first step is ... **Deng Yaping won the women's singles table tennis gold medals at the 1992 Barcelona Olympics and the 1996 Atlanta Olympics.** ... great work. In practice you don't have to worry much about the third criterion. Ambitious people are if anything already too ... **Deng Yaping was born in 1973** ...

Question: In which year was the Atlanta Olympics women's singles table tennis champion born?

Figure 4: *Multi-needle* setting in our PLUS version.

comprehensively assesses a model’s performance on inputs of different lengths and with critical information positioned at various locations.

Based on this setup, we propose the *Needle-in-a-Haystack PLUS* benchmark. We have made upgrades in three aspects: task difficulty, data diversity, and prevention of data leakage:

(1) Task difficulty: The original *Needle-in-a-Haystack* constitutes a simple QA task, where one needle corresponds to one question and the answer to the question is explicitly stated in the needle. We name this task as *single-needle QA* task. It emphasizes evaluating a model’s ability to retrieve relevant information.

To better assess a model’s reasoning capabilities on long documents, we introduce the *multi-needle QA* task, where multiple needles correspond to one question and the answer to the question should refer to more than one needles. For instance, to answer the question in Figure 4, the model must first identify the key entity Deng Yaping from Needle 1, and then infer based on this information that the year mentioned in Needle 2 is the answer

to the original question.

(2) **Data diversity:** In the original *Needle-in-a-Haystack*, for different haystack length L and depth percentages α , only a single needle about San Francisco and its corresponding question (see Figure 3 are provided for evaluation. This could introduce substantial evaluation bias. In contrast, we collect 100 needles from SQuAD dataset for *single-needle* QA and 60 pairs of needles from HotpotQA for *two-needle* QA. Given a haystack length L and a needle position α , we randomly select 10 needles from the 100 needles for single-needle evaluation (select 10 pairs of needles from the 60 pairs of needles for multi-needle evaluation), and report the average performance across 10 runs in our experiments. **It is important to note that there is no overlap between any of the evaluation data and the agent training data described in Section 2.5.**

(3) **Prevention of data leakage:** LLMs have undergone extensive pre-training and have amassed a wealth of global knowledge. Therefore, the model may directly respond based on the knowledge encoded in its parameters, rather than find the answer from the needle. To more accurately reflect the model’s long-document QA capability, we deliberately alter the needle to be unrealistic and expect the model to provide an answer that aligns with the needle, rather than one that conforms to the facts. For example, we may change a needle from “Deng Yaping won the women’s singles table tennis gold medals at the 1992 Barcelona Olympics”, a real fact, to “Deng Yaping won the women’s singles table tennis gold medals at the 2020 Tokyo Olympics”, a wrong statement, and expect the model to tell us “2020” for the question “In which year did Deng Yaping win the women’s singles table tennis gold medals in the Olympics?”.

Other Existing Benchmarks: LongBench (Bai et al., 2023) and InfiniteBench (Zhang et al., 2023) are two widely used long-text evaluation benchmarks currently. We introduce all the QA-related tasks from these two benchmarks for our evaluation. Specifically, LongBench includes 5 long document QA tasks: NarrativeQA, Qasper, Musique, HotpotQA, and 2wikimqa. The document lengths in these tasks range from 0 to 40,000 tokens. InfiniteBench includes one QA task, which is the Fake Book QA task. The document lengths in this task range from 0 to 200,000 tokens.

3.2 Evaluation Metrics

For all the evaluation tasks employed in this paper, we use `accuracy` as the evaluation metric. We recruit three undergraduate students majoring in language-related disciplines from top universities to serve as evaluators. They are tasked with assessing the semantic consistency between the model outputs and the ground truth answers, focusing solely on whether they align factually rather than considering differences in phrasing or other minor details.

3.3 Baselines

PI (Chen et al., 2023b). Extending the context window sizes of RoPE-based pretrained large language models by position interpolation.

Retrieval-Augmented Generation (RAG). RAG combines retrieval models, designed for searching large datasets or knowledge bases, with generation models. We implement RAG based on BGE m3 (Chen et al., 2024) model.

Claude2.1 (Anthropic, 2023). The Claude 2.1 released by Anthropic Corporation features a context window of 200K tokens and has significantly reductions in rates of model hallucination.

GPT-4 Turbo (OpenAI, 2023). The GPT-4 Turbo model from OpenAI offers a context window of 128K and can process text exceeding 300 pages within a single prompt.

In Figures 10 and 11 in the Appendix, we additionally provide more baselines for comparison, including Yarn (Peng et al., 2023), and the RAG methods based on TF-IDF and BM25 retrievers.

4 Results and Discussion

4.1 Overall Performance

To demonstrate the superiority of LONGAGENT in handling long texts, we compare it against powerful commercial models GPT-4 Turbo and Claude 2.1, as well as the most popular academic methods for long-text processing, PI and RAG (BGE M3).

Through multi-agent collaboration, LLaMA with a 4k-long context window is able to handle QA of up to 128k-token-long documents.

We fine-tune LLaMA2-7B using the method described in Section 2.5 to construct the leader and member agents. The results for *Single-Needle QA* and *Multi-Needle QA* in the *Needle-in-a-Haystack PLUS* are shown in Figure 5 and 6, respectively. Comparing the subfigures titled OpenAI GPT4 128k and LongAgent (fine-tuned

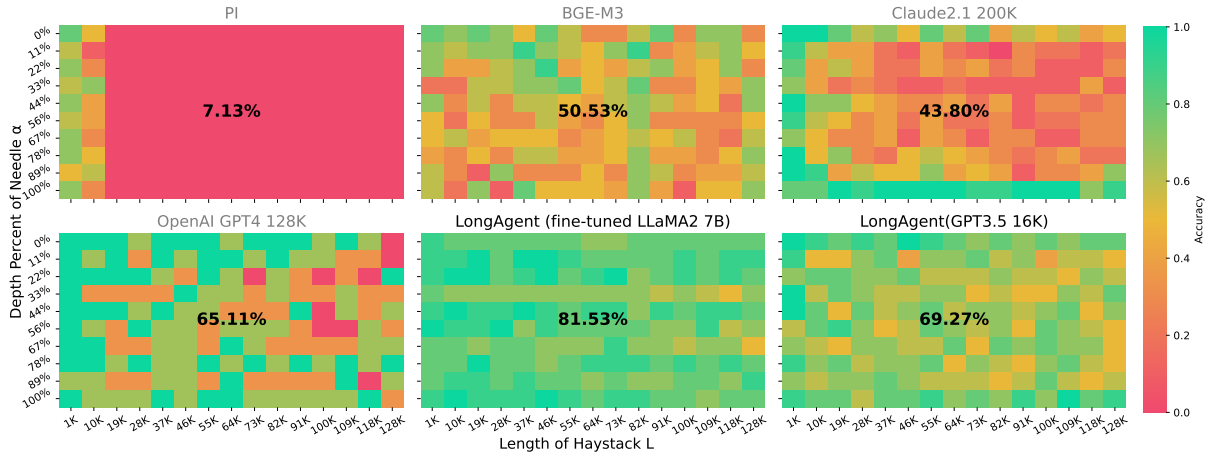


Figure 5: Single-Needle QA Results with *Needle-in-a-Haystack PLUS*. With the help of LONGAGENT, LLaMA2-7B achieves an average accuracy improvement of 16.42% compared to GPT-4 across the range from 1k to 128k (increasing from 65.11% to 81.53%). The bold black numbers in each subfigure indicate the average accuracy.

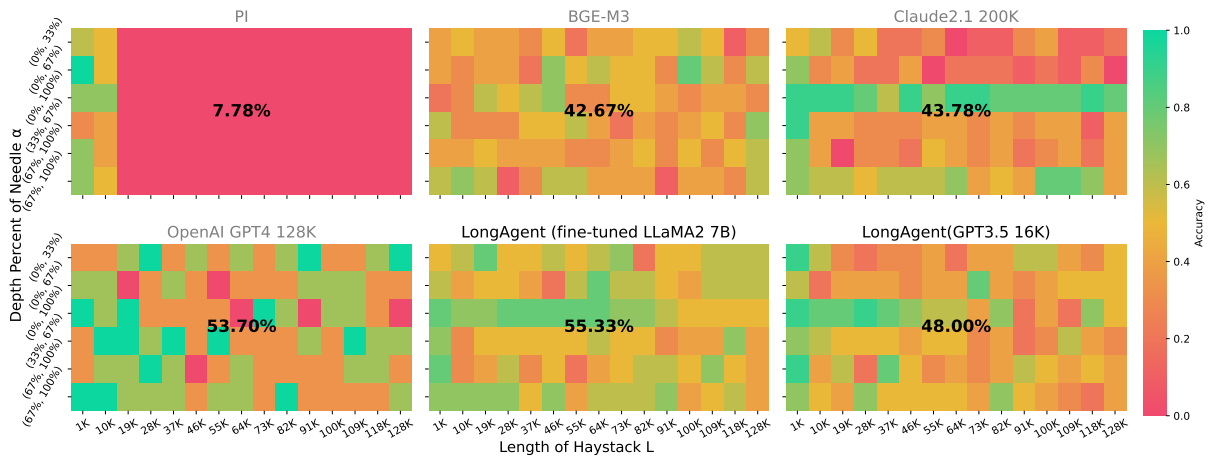


Figure 6: Multi-Needle QA Results with *Needle-in-a-Haystack PLUS*. With the help of LONGAGENT, LLaMA2-7B model achieves an average accuracy improvement of 1.63% compared to GPT-4 across the range from 1k to 128k (increasing from 53.70% to 55.33%). The two percentage values on the y-axis represent the depth percentages of Needle 1 and Needle 2 respectively. The bold black numbers in each subfigure indicate the average accuracy.

LLaMA2 7B), we find that LONGAGENT outperforms GPT-4 across haystack length ranging from 1k to 128k, with an average improvement of 16.42% and 1.63% on Single-Needle QA and Multi-Needle QA respectively.

Additionally, we fine-tune PI and YaRN using QA data of lengths 1k to 16k (training on longer contexts led to OOM errors on our GPUs). From the subplots titled PI in Figures 5 and 6, and the subplots titled YaRN in Figures 10 and 11 in the appendix, we can see that these methods fail to effectively extrapolate to the 19k-128k input range, while LongAgent’s accuracy does not degrade within the 128k range as input length increases, demonstrating its advantage in extrapolation ability. Additionally, we compared

against the RAG methods supported by tf-idf, bm25, and BGE m3 retrievers, and LongAgent consistently outperforms them as well.

For aligned LLMs like GPT-3.5, LongAgent can work without fine-tuning.

As described in Section 2.5, we prompted GPT-3.5 to play the roles of the leader and member agents for long document QA. Although GPT-3.5 only has a 16k context window, LONGAGENT allows it to handle inputs far exceeding 16k tokens length out-of-the-box. Comparing the subfigure titled OpenAI GPT4 128k and LongAgent (GPT3.5 16k) in Figure 5, we find that GPT-3.5 achieves a 6.78% accuracy improvement on Single-Needle QA. In the multi-needle setting shown in Figure 6, it also matches GPT-4’s performance.

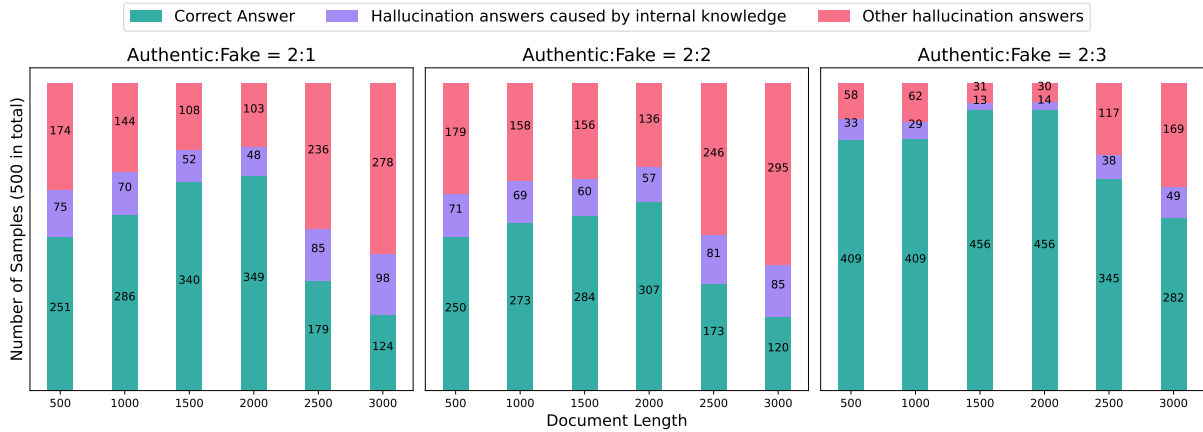


Figure 7: The influence of training data recipe on model hallucinations. Each piece of data is a 3k-token-long document paired with a question. The data is considered *authentic* if the question can be answered using the document. Conversely, it is deemed *fake* if the document lacks content relevant to the question. We employ various data recipes to train the model, adjusting the ratio of authentic to fake data (2:1, 2:2, and 2:3). We assess the model’s tendency to produce hallucinated responses by using fake data, with document lengths varying between 500 and 3000 tokens. For each specific document length, we evaluate using 500 pieces of fake data. Ideally, the model should consistently respond with *No Mention* to all 500 items. However, it may also generate hallucinated answers based on its internal knowledge or other factors.

Benchmark	Tasks	GPT-4	LONGAGENT (GPT3.5 16k)
LongBench	NarrativeQA	0.600	0.680
	Qasper	0.560	0.620
	Musique	0.460	0.400
	HotpotQA	0.540	0.520
	2wikimqa	0.540	0.560
InfiniteBench	FakeBookQA	0.500	0.520

Table 1: Performance comparisons on more long-document QA tasks. Considering evaluation cost of GPT-4, we randomly selected 50 samples per task for evaluation. LONGAGENT (GPT3.5 16k) outperforms GPT-4 on more than half of the tasks. This result is non-trivial, as GPT-3.5’s reasoning capability is weaker than GPT-4’s, and its context window (16k) is much smaller than GPT-4’s (128k)

In Table 1, we also test the QA-related tasks in LongBench and InfiniteBench. The results show that LONGAGENT(GPT3.5-16k) outperforms GPT-4 on more than half of the tasks. Given that the original context window of GPT-3.5 is 16k, far smaller than the 128k of GPT-4, and that the capability of GPT-3.5 itself is weaker than GPT-4, the above results are sufficient to demonstrate the effectiveness of LONGAGENT.

4.2 Hallucination Analysis

We find that LONGAGENT’s errors mainly stem from a specific type of member hallucination:

when the assigned text chunk does not contain information relevant to the leader’s query, the member sometimes fabricates an response. This section investigates two key factors, the recipes in the training data and the document length (i.e., the length of the text chunk assigned to the member) on member hallucination. Comparing the three subfigure in Figure 7, as the proportion of *fake* type data in the member’s training data increases (from 2 : 1 to 2 : 3), the percentage of correctly answering *No Mention* significantly improves. Particularly when *authentic:fake* reaches 2 : 3, in the 4 test document length groups from 500 to 2,000, the number of correctly answered documents exceeds 400 in each group.

Meanwhile, the length of the input document is also an important factor affecting member hallucination. The three subfigure in Figure 7 all show the following trend: as the document length increases from 500 to 2,000 tokens, the number of samples receiving correct responses gradually increases. This is mainly because the member’s training data consists of 3,000-token-long documents. The increase in test document length gradually reduces the length gap between training and test data. However, when the test document length exceeds 2,000, the member’s degree of hallucination starts to increase. We speculate that this may be because the input document length is gradually approaching LLaMA2’s pretraining length (4k),

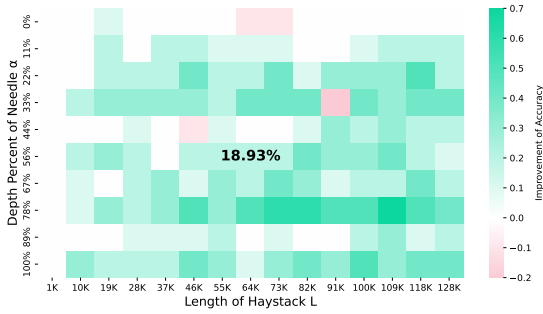


Figure 8: Improved accuracy through *inter-member communication* mechanism. ‘18.93%’ denotes the average Acc improvement across different α and L .

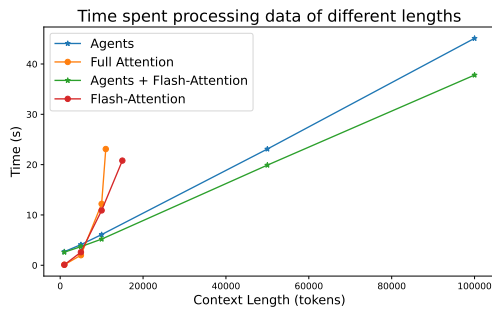


Figure 9: LONGAGENT scheme exhibits significantly superior time and memory efficiency compared to directly perform full attention on long texts.

and LLaMA2’s own sequence modeling capability becomes increasingly inadequate at such lengths.

4.3 Ablation Study

In this section, we verify the effectiveness of the cross-member communication mechanism in mitigating member hallucination. As shown in Figure 8, after introducing the *cross-member communication mechanism*, almost all the cells are green. This means that the *cross-member communication mechanism* achieved positive accuracy improvements under different settings of haystack length and needle depth percentage (18.9% accuracy improvement on average). Furthermore, the number of members increases with the length of the text, and the number of members experiencing hallucinations also grows. In this context, the improvement in accuracy brought about by conflict resolution becomes even more evident.

4.4 Efficiency Advantage

Thanks to chunking of long texts, LONGAGENT’s time complexity for processing long texts is $\mathcal{O}(N)$. In this subsection, we empirically verify this point. As shown in Figure 9, the latency of

LONGAGENT within the range of $1k$ - $100k$ almost grows linearly with length. For Full Attention, which has quadratic complexity, the inference latency increases rapidly regardless of the use of techniques such as flash attention. The latency of Full Attention when processing $10k$ tokens has already exceeded that of LONGAGENT processing $50k$ tokens. Furthermore, without specific memory optimization techniques, a single A100 GPU with 80G memory can only support text inference up to $11k$ in length, and even with flash attention, this number can only be increased to $15k$. Under the same settings, LONGAGENT can process contexts of around $100k$ with less than 40G of memory.

5 Related Works

Several methods have been proposed to extend the positional encoding (PE) for handling longer sequences. Initially, approaches like RoPE and PI (Chen et al., 2023b) attempted to interpolate position indices within pre-trained limits, but neglected frequency variations. Recent advancements include "NTK-aware" (Bloc97, 2023a) interpolation and "Dynamic NTK" (Bloc97, 2023b) interpolation, which address high-frequency component losses. Additionally, "NTK-by-parts" (Bloc97, 2023c) interpolation outperforms others when fine-tuned on longer-context data. Another popular approach for managing longer sequences involves constraining global causal attention to local attention. ReRoPE (Su, 2023) truncates context lengths during pretraining and LM-Infinite (Han et al., 2023) restricts attention to a chevron-shaped window. Mohtashami and Jaggi (2023) insert landmark tokens after text fragments, while Zhang et al. (2024) propose beacon tokens for summarizing fragments.

6 Conclusions

This paper proposes LONGAGENT, a novel long-document QA approach based on multi-agent collaboration. The proposed *inter-member communication* mechanism alleviates the member hallucination when they reading documents, thus facilitating effective management by the leader of dozens to hundreds of members. We have also developed *Needle-in-a-Haystack Plus* to facilitate a comprehensive assessment of the LLM’s understanding on long documents. Our experimental results indicate that LONGAGENT offers a promising alternative for long-document QA.

542 Limitations

543 The direction of long text processing based on
544 multi-agent collaboration still has many interesting
545 points yet to be explored: (1) More diverse types
546 of agents and broader task scopes: Current agents
547 already possess capabilities such as tool invocation,
548 coding, and multimodal understanding. Therefore,
549 an intriguing question is how to fully leverage
550 these capabilities of agents and handle long
551 sequences of document summarization, repository-
552 level code processing, video understanding, and
553 so on. (2) Further alleviating hallucinations:
554 Reducing hallucinations during the multi-agent
555 collaboration process is crucial for the ultimate
556 collaborative effect. Although this paper proposes
557 a mechanism to mitigate hallucinations, further
558 research on this issue is still highly necessary.

559 References

560 Anthropic. 2023. Model card and evaluations for claude
561 models. Website. <https://www.anthropic.com/product>.
562

563 Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu,
564 Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao
565 Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang,
566 and Juanzi Li. 2023. [Longbench: A bilingual, multitask benchmark for long context understanding](#).
567

568 2023a Bloc97. 2023a. Ntk-aware scaled rope
569 allows llama models to have extended (8k+)
570 context size without any fine-tuning and
571 minimal perplexity degradation. [https://www.reddit.com/r/LocalLLaMA/
572 comments/141z7j5/ntkaware_scaled_
573 rope_allows_llama_models_to_have/](https://www.reddit.com/r/LocalLLaMA/comments/141z7j5/ntkaware_scaled_rope_allows_llama_models_to_have/).
574

575 2023b Bloc97. 2023b. Dynamically scaled rope
576 further increases performance of long context
577 llama with zero fine-tuning. [https://www.reddit.com/r/LocalLLaMA/
578 comments/14mrgpr/dynamically_
579 scaled_rope_further_increases/](https://www.reddit.com/r/LocalLLaMA/comments/14mrgpr/dynamically_scaled_rope_further_increases/).
580

581 2023c Bloc97. 2023c. Ntk-aware interpolation "by
582 parts" correction, 2023. URL <https://github.com/jquesnelle/scaled-rope/pull/1>.
583

584 Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong
585 Liang, and Lidong Bing. 2023a. [Clex: Continuous length extrapolation for large language models](#).
586

587 Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu
588 Lian, and Zheng Liu. 2024. [Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation](#).
589
590

591 Shouyuan Chen, Sherman Wong, Liangjian Chen, and
592 Yuandong Tian. 2023b. [Extending context window of large language models via positional interpolation](#).
593

594 Yukang Chen, Shengju Qian, Haotian Tang, Xin
595 Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023c. [Longlora: Efficient fine-tuning of long-context large language models](#).
596
597

598 Chi Han, Qifan Wang, Wenhan Xiong, Yu Chen, Heng
599 Ji, and Sinong Wang. 2023. [Lm-infinite: Simple on-the-fly length generalization for large language models](#).
600
601

602 Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman,
603 Shantanu Acharya, Dima Rekish, Fei Jia, Yang
604 Zhang, and Boris Ginsburg. 2024. [Ruler: What's the real context size of your long-context language models?](#)
605
606

607 Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng
608 Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen,
609 and Xia Hu. 2024. [Llm maybe longlm: Self-extend llm context window without tuning](#).
610

611 Greg Kamradt. 2023. Needle in a haystack - pressure
612 testing llms. Website. [https://github.com/
613 gkamradt/LLMTest_NeedleInAHaystack](https://github.com/gkamradt/LLMTest_NeedleInAHaystack).

614 Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin
615 Paranjape, Michele Bevilacqua, Fabio Petroni, and
616 Percy Liang. 2023. [Lost in the middle: How language models use long contexts](#).
617

618 Amirkeivan Mohtashami and Martin Jaggi. 2023. [Landmark attention: Random-access infinite context length for transformers](#).
619
620

621 OpenAI. 2023. [Gpt-4 technical report](#).
622

623 Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico
624 Shippole. 2023. [Yarn: Efficient context window extension of large language models](#).
625

626 Ofir Press, Noah A. Smith, and Mike Lewis. 2022. [Train short, test long: Attention with linear biases enables input length extrapolation](#).
627

628 Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and
629 Percy Liang. 2016. [SQuAD: 100,000+ questions for machine comprehension of text](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–
630 2392, Austin, Texas. Association for Computational
631 Linguistics.
632
633

634 Jianlin Su. 2023. Rectified rotary position embeddings.
635 <https://github.com/bojone/rerope>.
636

637 Szymon Tworkowski, Konrad Staniszewski, Mikołaj
638 Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr
639 Miłoś. 2023. [Focused transformer: Contrastive training for context scaling](#).
640

641 Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song
642 Han, and Mike Lewis. 2023. [Efficient streaming language models with attention sinks](#).
643

644 Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao,
645 Qiwei Ye, and Zhicheng Dou. 2024. [Soaring from 4k to 400k: Extending llm's context with activation beacon](#).
646
647

648 Xinrong Zhang, Yingfa Chen, Shengding Hu, Qihao Wu,
649 Junhao Chen, Zihang Xu, Zhenning Dai, Xu Han,
650 Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2023.
651 Infinitebench: 128k long-context benchmark for
652 language models.

653 Wangchunshu Zhou, Yuchen Eleanor Jiang, Peng
654 Cui, Tiannan Wang, Zhenxin Xiao, Yifan Hou,
655 Ryan Cotterell, and Mrinmaya Sachan. 2023.
656 [Recurrentgpt: Interactive generation of \(arbitrarily\)
657 long text.](#)

658 Appendix

659 A Additional Results

660 Figures 10 and 11 provide additional experimental
661 results of more baselines on the large document
662 retrieval benchmark.

663 B Prompt Used in Multi-agent 664 Collaboration

665 Table 2-4 presents the prompt templates used in the
666 multi-agent collaboration process.

You are the leader of a team of {member_nums}
members. Your team will need to collaborate to
solve a task. The rule is:

1. Only you know the task description and task objective; the other members do not.
2. But they will receive different documents that may contain answers, and you need to send them an instruction to query their document.
3. Your instruction need to include your understanding of the task and what you need them to focus on. If necessary, your instructions can explicitly include the task objective.
4. Finally, you need to complete the task based on the query results they return.

Task Description:

Answer the question based on the given passages. The answer must be extracted from the given passages.

Task Objective:

{user_question}

Generate Instruction for Members:

Now, you need to generate an instruction for all team members. You can ask them to answer a certain question, or to extract information related to the task, based on their respective documents. Your output must following the JSON format: `{{"type": "instruction", "content": "your_instruction_content"}}`

Table 2: Prompt templates for the Leader to understand user queries and generate instructions to members

Document:

{member_chunk}

Instruction:

Answer the question based on the given passages. The answer must be extracted from the given passages. Question: {Leader_Instruction}

You are an experienced reader; please summarize the content in the document related to the instructions in a <scratchpad> tag, then describe your response." Your output must following the JSON format: `{{"type": "response", "content": "your_response_content"}}`

The "content" needs to be as concise as possible.

Table 3: Prompt template for Member to read text chunks and respond to Leader instructions

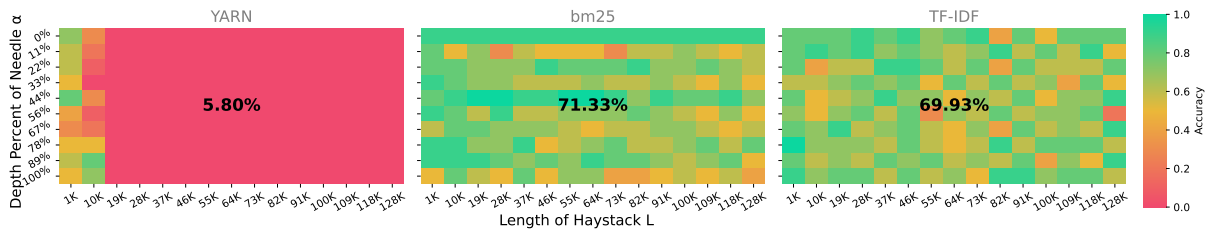


Figure 10: Addition Results of Needle-in-a-Haystack Plus in Single-Needle QA Setting

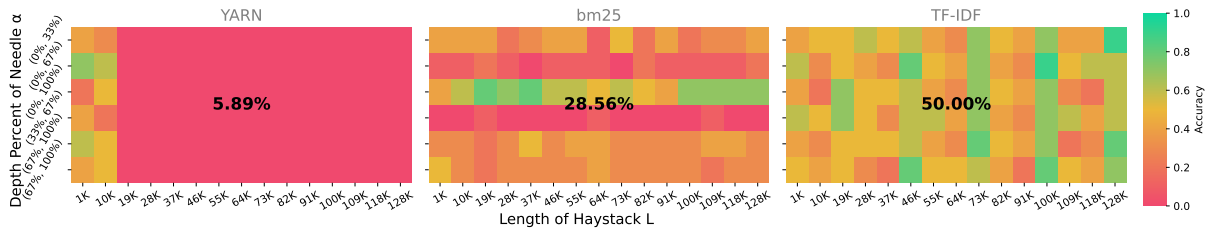


Figure 11: Addition Results of Needle-in-a-Haystack Plus in Multi-Needle QA Setting

Here are the responses from all the members. Each member sees different segments of a document, and these segments do not intersect with each other. The correct answer may appear in any one or several members' responses.

Note that if a minority of members find information relevant to the question while the majority reply that the document does not contain information relevant to the question, you should pay attention to the replies from those members who found relevant information.

Member Response:

{Member_Response}

Task Description:

Answer the question based on the given passages. The answer must be extracted from the given passages.

Task Objective:

{User_Question}

Determination:

Based on the above information, you need to determine if you can solve the task objective. You have two choices:

1. If members' responses cannot solve the task objective, or if their responses contain conflicting answers, provide a new instruction for them to answer again.
2. Else, if the task objective can be solved, give your final answer as concisely as you can, using a single phrase if possible. Do not provide any explanation.

Your output must following the JSON format: `{{"type": "answer", "content": "your_answer_content"}}` or `{{"type": "instruction", "content": "your_instruction_content"}}`

Table 4: Prompt template for Leader to make decisions and generate new instructions.