Chain of Agents: Large Language Models Collaborating on Long-Context Tasks

Yusen Zhang[♣]*, **Ruoxi Sun**[♢], **Yanfei Chen**[♢], **Tomas Pfister**[♢], **Rui Zhang**[♣†], **Sercan Ö. Arık**[♢†]
[♣] Penn State University, [♢] Google Cloud AI Research
{yfz5488, rmz5227}@psu.edu, {ruoxis, yanfeichen, tpfister, soarik}@google.com

Abstract

Addressing the challenge of effectively processing long contexts has become a critical issue for Large Language Models (LLMs). Two common strategies have emerged: 1) reducing the input length, such as retrieving relevant chunks by Retrieval-Augmented Generation (RAG), and 2) expanding the context window limit of LLMs. However, both strategies have drawbacks: input reduction has no guarantee of covering the part with needed information, while window extension struggles with focusing on the pertinent information for solving the task. To mitigate these limitations, we propose *Chain-of-Agents (CoA)*, a novel framework that harnesses multi-agent collaboration through natural language to enable information aggregation and context reasoning across various LLMs over long-context tasks. CoA consists of multiple worker agents who sequentially communicate to handle different segmented portions of the text, followed by a manager agent who synthesizes these contributions into a coherent final output. CoA processes the entire input by interleaving reading and reasoning, and it mitigates long context focus issues by assigning each agent a short context. We perform a comprehensive evaluation of CoA on a wide range of long-context tasks in question answering, summarization, and code completion, demonstrating significant improvements by up to 10% over strong baselines of RAG, Full-Context, and multi-agent LLMs.

1 Introduction

Despite their impressive performance across a wide range of scenarios, LLMs struggle with tasks that involve long contexts [8, 63, 57]. Numerous application scenarios demand extremely long contexts, such as question answering [85, 22, 69], document and dialogue summarization [25, 93, 91, 90, 12], and code completion [20, 43], where the inputs contain entire books [32, 33] and long articles [16].

To tackle the challenge with long context tasks, two major directions have been explored as shown in Table 1: *input reduction* and *window extension*. *Input reduction* reduces the length of the input context before feeding to downstream LLMs. Truncation approaches [1, 67] directly truncate the input. Retrieval Augmented Generation (RAG) [81] extends this direction by retrieving the most relevant chunks through embedding similarity. However, because of low retrieval accuracy, LLMs could receive an incomplete context for solving the task, hurting performance. *Window extension* extends the context window of LLMs via finetuning to consume the whole input [13, 44, 48]. For example, Claude-3 [5] directly allows reading 200k tokens for each input. However, when the window becomes longer, LLMs struggle to focus on the needed information to solve the task, suffering from ineffective context utilization such as the *"lost in the middle"* issue [37, 3, 42].

*Work done while the author was a student researcher at Google Cloud AI Research.
†Last authors

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

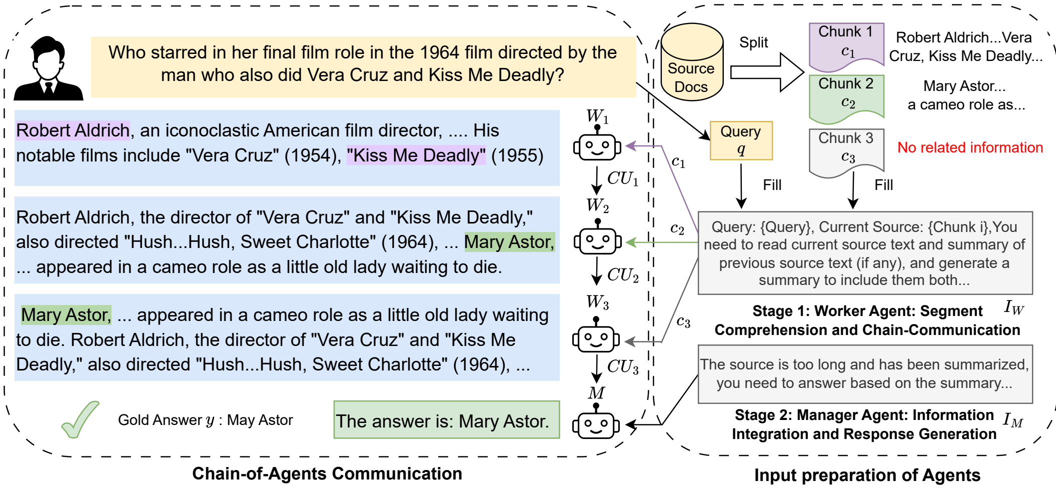


Figure 1: Overview of Chain-of-Agents, a training-free, task agnostic, and highly-interpretable framework that harnesses multi-agent collaboration for long-context tasks. Blue boxes indicate communication unit CU_i between worker agents W_i and W_{i+1}. It consists of multiple worker agents who sequentially communicate to handle different segmented portions of the text, followed by a manager agent who synthesizes these contributions into a coherent final output.

Table 1: Comparison between Chain-of-Agents and prior methods for long-context tasks. Rec./Foc.: being able to mitigate inaccurate receptive field/long context focusing issues. Read: the number of tokens as model input, where n is the total input length, k is the context window limit of LLMs. Inter.: the interpretability of the approach. Note that RAG is 'medium interpretable' because of the re-ranked chunks.

Category	Example Work	Rec.	Foc.	No Train	Read	Agent	Applicability	Inter.
Input Reduction	Truncation [50]	✗	✓	✓	k	Single	Generic	Low
	RAG [81]	✗	✓	✗	$n+k$	Single	Query-based	Medium
Window Extension	Position Interpolation [13]	✓	✗	✗	n	Single	Generic	Low
	Long Context [5]	✓	✗	✗	n	Single	Generic	Low
Multi-agent LLMs	Chain-of-Agents (Ours)	✓	✓	✓	n	Multiple	Generic	High

Motivated by the aforementioned challenges, we propose a novel framework, *Chain-of-Agents (CoA)*, inspired by the way humans interleave reading and processing of long contexts under limited working memory constraints of brains [15]. The key idea of CoA is to harness multi-agent communication to enable information aggregation and context reasoning capabilities across different LLMs. As shown in Figure 1, CoA contains two stages. In stage 1, a series of worker agents in charge of different chunks of long context collaborate and aggregate evidence for answering the given query. To this end, the workers read and process sequentially, each receiving the message from previous worker and transferring the useful updated information to the next. In stage 2, the manager agent receives the complete evidence from last worker agent and generates the final response.

As shown in Table 1, CoA is a training free, task agnostic, and highly interpretable framework processing entire "receptive field" by interleaved reading-processing and mitigating the long context focusing issue by assigning each agent a short context. Different from *input reduction* where LLMs need to start processing with low receptive field over reduced inputs ("read-then-process"), workers in CoA start to process each chunk before reading all input ("interleaved read-process"), tackling the problems that input reduction struggles with, such as, generic summarization or counting of passages [6]. Different from *context extension*, CoA leverages the capability of communication rather than trying to feed many tokens into an LLM. This is a more natural solution for complex tasks because we assume that each LLM has its limit and there are always complex context tasks surpassing its limit. Compared with Full-Context, CoA is also cost effective by reducing time complexity from n^2 to nk, where n is input tokens and k is the context limit of LLMs.

We conduct intensive experiments on *nine datasets*, including question answering, summarization, and code completion tasks with *six LLMs*, with PaLM 2 [4], Gemini [67], and Claude 3 [5] models. We compare CoA with two strong baselines chosen from *input reduction* and *window extension* approaches, respectively: (i) RAG, which uses a state-of-the-art retriever to obtain the most relevant information to feed into the LLM and (ii) Full-Context (Vanilla), which feeds all input into the LLM until reaching the window limit. Our results show that on all nine datasets, CoA obtains significant improvement over all baselines by up to 10%. Noting that there is not enough research on multi-agent for long context tasks, we carefully create two multi-agent baselines, including a hierarchical structure and result merging approach to further demonstrate that CoA is superior among other possible multi-agent frameworks.

2 Related work

Multi-agent LLMs. Multi-agent LLMs has become a popular topic [21]. A large proportion of works focus on social simulation [52, 53] and frameworks [35]. Based on the success of them, some works explore the game settings [39, 74, 82, 83, 47], world wars [23], economy markets [36, 76], recommendation systems [88], and pandemics [19]. Others advance problem solving, focusing on reasoning of short text via multi-agent debating [18, 80, 10, 66] and discussing [11, 58] for different tasks in reasoning [18, 66], mechanics problems [49], paper review [84], knowledge graph construction [86], and code intelligence [70, 24]. Different from the above works, we improve problem-solving on long context tasks. To the best of our knowledge, the closest work LongAgent [92] utilizes a tree structure to do multi-hop QA over a long input context and the sibling worker agents do not communicate, while CoA utilizes a chain structure allowing the communication units to flow between workers (detailed comparison in Appendix G and F.3).

Long Context Modeling for LLMs. *Input Reduction*: RAG is broadly leveraged to solve long context query-based tasks [81, 2]. Combined with a strong retriever [79, 41, 73], LLMs are expected to handle long context questions in open domains. Previous studies have augmented LLMs during pretraining [26, 72], finetuning [34], inference [87], or directly integrating [28, 61]. Moreover, some token-level retrieval approaches are proposed [38]. Longllmlingua [27] removes tokens from long prompt to compress long context prompt to a desired budget. *Window Extension*: The context windows of LLMs are getting longer and longer thanks to the development of GPUs. For instance, the context window increases from 1024 (GPT-2 [56]), 2048 (GPT-3 [7]), to 128k (GPT-4 [50]). Moreover, the newest version of Claude-3 [5] supports 200k context windows. To save the cost of LLM training, some continual learning or finetuning approaches are proposed to extend the context window of pretrained LLMs [46, 55, 44, 48]. For instance, position interpolation [13] modifies rotary position encoding [64] and extends the context length of LLaMA [68] to 32k. Different from the above works, CoA does not reduce the input length or extend the window length of LLMs, but rather leverages multi-agent collaboration and communication to obtain the full receptive field.

Recently, a few studies have employed text chunking algorithms to divide and process lengthy text inputs. Notably, RecurrentGPT [95] adopts a chain structure to generate long outputs in a chunk-by-chunk manner, maintaining continuity in the storyline. WalkMaze [9] utilizes a centralized tree structure without communication between sibling agents to do single-hop QA over a long input context. Different from these works, CoA focuses on long input tasks with a chain structure with inter-sibling communication (Append G).

Complex Task Reasoning. Previous works on complex reasoning have focused on decomposing the complex question into sub-questions to solve them step-by-step. [54] decompose the questions with an unsupervised model and answer them separately with another model. Decomposed Prompting [30] leverages some predefined modules to classify each decomposed sub-question, then further decompose if needed. Additionally, decomposing is used for human-computer interaction [77], and prompter training [71]. Recently, many work has been proposed for LLMs, such as Chain-of-thought [75] Least-to-most prompting [94] and Pearl [65]. However, the length of the prompt does not exceed the context limit of a single agent. By contrast, our Chain of Agents framework is proposed to effectively reason across multiple agents to support the unlimited length of source text.

3 Method

Figure 1 shows the overview of our Chain-of-Agents (CoA) framework, containing two stages. In stage 1, long context is split into chunks where each chunk can be processed by a worker agent. Then, the worker agents communicate sequentially to produce evidence over the entire context. In stage 2, a manager agent consumes the knowledge from the chain of workers to generate the final answer.

To formulate the task, we denote a long-context sample as (x, y, q), where x is the input of n tokens, y is the output of m tokens, q is an optional query. Given an LLM with k tokens (usually $k \ll n$) as the context window limit, the target is to generate y with the limited input context window. Therefore, we divide each source text x into chunks $x = \{c_1, c_2 ... c_l\}$, so that each chunk can be completely fed into the LLM agent backbone model.

Algorithm 1 Chain of Agents (CoA).

Input: Source input x, query q, agent window size k, large language model LLM$(*)$.
Output: Answer to the query.
Split x into l chunks $\{c_1, c_2, \cdots, c_l\}$ where c_i is shorter than k
Initialize $CU_0 \leftarrow$ empty string.
for i in $1, 2, \cdots, l$ **do**
  $CU_i \leftarrow \text{LLM}_{W_i}(I_W, CU_{i-1}, c_i, q)$
end for
return $\text{LLM}_M(I_M, CU_l, q)$

3.1 Stage 1: Worker Agent: Segment Comprehension and Chain-Communication

In Stage 1, CoA contains a sequence of l number of worker agents. Each worker W_i inputs the concatenation of a chunk c_i from source text x, a query q, instruction for a specific task for worker agent I_W, and the message passed from the previous agent, denoted as "communication unit" CU_{i-1}. The communication is unidirectional, passing from worker agent $i-1$ to i. The worker agents process them and output the message CU_i for next worker, expressed as:

$$CU_i = \text{LLM}_{W_i}(I_W, CU_{i-1}, c_i, q), \tag{1}$$

CUs produced by worker agents vary across different tasks. For question answering, CU contains the evidence for the manager to answer the question. For summarization, CU contains the summary of the previous texts. For code completion, CU contains the code summary with function/class names and explanation. Effectiveness on diverse tasks demonstrates the flexibility of CoA (Appendix C).

The multi-step worker communication in CoA expands the model context to the full receptive field, meaning that the last worker can read the full input no matter how long the input is. Therefore, CoA is extensible to inputs with different lengths by adjusting the number of worker agents.

The left side of Figure 1 underscores the necessity of collaborative communication among workers to effectively address complex, long-context reasoning tasks. We observe that 1) Although the question is unanswerable given c_1, W_1 generates related evidence that is useful for answering the question; 2) with the partial answer from the previous worker, W_2 further reasons with the current source to complete the full reasoning chain across agents and generate the interpretative reasoning chain; 3) W_3 finds no related information in the chunk 3, it directly rewrites CU_2 by putting the correct answer as the first token of CU_3 without adding any unrelated information (Appendix F.4). This shows that if workers are independent (such as tree structure communication), it is impossible to answer hop two while the answer of hop one is held by another worker (Appendix G).

3.2 Stage 2: Manager Agent: Information Integration and Response Generation

In Stage 2, after multiple steps of information extraction and comprehension by worker agents, the manager agent produces the final solution. While worker agents extract relevant information in a long-context source, the manager agent synthesizes relevant information accumulated by the end of *"worker-agent-chain"* to generate the final answer. Specifically, given the instruction for manager I_M and query q, the manager agent consumes accumulated knowledge from last worker CU_l and generates the final answer $Response$:

$$Response = \text{LLM}_M(I_M, CU_l, q) \tag{2}$$

The benefit of using a separate LLM as the manager agent is to decompose the duty of analyzing chunks in the long-context source (*"worker agents"*) and producing the final answer (*"manager agent"*), so that every agent can fulfill its duty to the most[3].

[3]Other design choices: Our experiments show that using the last worker W_l to directly generate the final result leads to a performance drop. Besides, feeding the manager with all CU_i or some CU that is related to the answer (decided by W_i) also hurts the performance because of confusion led by conflicting CU_i.

Table 3: **Dataset Statistics**. Avg. Input/Agents is the average words/agents (8k) for source input.

	Question Answering					Summarization			Code
	HotpotQA	MuSiQue	NarrativeQA	Qasper	QuALITY	QMSum	GovReport	BookSum	RepoBench-P
Avg. Input	10603	12975	71787	4236	4936	12524	9239	108478	7105
Avg. Agents	2.35	2.88	12.45	1.12	1.31	2.57	2.03	18.63	1.69
Query-based	✓	✓	✓	✓	✓	✓	✗	✗	✓

3.3 Time Complexity Analysis

Table 2: Time complexity.

	Encode	Decode
Full-Context	$\mathcal{O}(n^2)$	$\mathcal{O}(nr)$
CoA	$\mathcal{O}(nk)$	$\mathcal{O}(nr)$
RAG	$\mathcal{O}(nk') + \mathcal{O}(k^2)$	$\mathcal{O}(n/k') + \mathcal{O}(kr)$

We compare the time cost of full-context input and Chain-of-Agents theoretically in a decoder-only setting. We assume the response generated by LLMs contains r tokens on average, the input has n tokens, the context limit of LLM is k, and the length of each chunk in RAG is k'. The time complexity is shown in Table 2 (Appendix A). As can be seen, the encoding time of CoA is less than Full-Context because $k \ll n$ in long context tasks, while they have the same decoding time. This demonstrates the efficiency of CoA compared with the Full-Context baseline.

4 Experiment

4.1 Experiment Setup

Datasets. We conduct experiments on nine long context datasets across three task types (Table 3):

- **Question Answering.** We consider five QA datasets from the LongBench [6] and SCROLL [60]. **HotpotQA** [85] is a Wikipedia-based multi-hop QA dataset. It requires reasoning across multiple passages to find the answer. **MuSiQue** [69] is a multi-hop QA dataset. It is much more difficult than HotpotQA as it contains more hops in one sample, unanswerable questions, and harder distracting content. **NarrativeQA** [31] is a QA dataset over entire books or movie transcripts. The answers can be abstract or extractive, yes/no, and unanswerable. **Qasper** [17] is a question answering dataset over NLP papers. It also contains extractive, abstractive, yes/no, and unanswerable questions. **QuALITY** [51] is a dataset based on stories and articles with multiple-choice questions for each sample. The model needs to select the correct answer among choices.
- **Summarization.** We pick two summarization datasets from SCROLLS. **QMSum** [93] is a query-based summarization dataset, formed by meeting transcripts from multiple domains such as academic and industrial products. **GovReport** [25] is a generic summarization dataset containing long reports published by the U.S. Government Accountability Office. We also use one dataset for long context memorization tasks. **BookSum** [33] is a collection of datasets for long-form narrative summarization, including novels, plays, and stories. We use the book-level partition of the BookSum dataset for experiments.
- **Code Completion.** We pick **RepoBench-P** [43] which is collected from GitHub repositories, and the model needs to generate the next line of code given the long code base.

Metrics. We report the geometric mean of ROUGE [40] for Summarization tasks, code similarity score [6] for Code Completion task, exact match for QuALITY dataset [60], and F1 score for the rest of the Question Answering datasets [6].

LLMs. We use six LLMs in total as the backbone of CoA across all experiments (Appendix B). **PaLM 2** [4] is a series of models with a dense left-to-right, decoder-only language model pretrained on a high-quality corpus of 780 billion tokens. We use **text-bison@001** and **text-unicorn@001** for the experiments with an 8k maximum context window. **Gemini 1.0** [67] is a family of LLMs proposed by Google. We use **gemini-ultra** for experiments. The input limit is 32k tokens for Gemini. **Claude 3** [5] is a family of large language models developed by Anthropic. The family includes three state-of-the-art models in ascending order of capability: **claude-3-haiku**, **claude-3-sonnet**, and **claude-3-opus**. These models are capable of consuming 200k tokens in the context window, providing a strong baseline for long context tasks. Although our framework is flexible to use diverse types of LLMs as workers and manager, we use the same model for each W_i and M if not specified.

Table 4: **Overall results of CoA**. CoA significantly outperforms Vanilla and RAG using various backbone LLMs on all datasets.

LLMs	Baselines	Question Answering					Summarization		Code
		HotpotQA	MuSiQue	NarrativeQA	Qasper	QuALITY	QMSum	GovReport	RepoBench-P
text-bison	Vanilla (8k)	45.57	26.87	11.96	26.56	61.86	15.45	20.60	56.30
	RAG (8k)	51.91	33.83	14.20	27.20	55.28	15.59	20.83	55.63
	CoA (8k)	**53.62**	**37.09**	**25.26**	**37.17**	**65.42**	**16.77**	**26.11**	**58.25**
text-unicorn	Vanilla (8k)	51.09	29.67	14.45	27.81	83.40	16.61	23.50	53.87
	RAG (8k)	58.01	40.38	19.12	24.44	83.00	16.83	21.43	50.49
	CoA (8k)	**62.04**	**42.49**	**20.37**	**38.01**	**83.80**	**17.67**	**26.48**	**60.39**
gemini-ultra	Vanilla (8k)	40.62	23.61	7.71	20.59	57.40	12.10	26.18	49.09
	Vanilla (32k)	45.09	27.93	7.21	21.71	58.60	10.24	26.96	73.04
	RAG (8k)	51.13	31.56	14.51	18.70	62.40	12.70	25.87	72.94
	CoA (8k)	**54.26**	**35.09**	**25.26**	**35.10**	**80.60**	**12.84**	**26.98**	**73.05**

Baselines. Our principle of choosing baselines is to find the strongest and most typical approaches from *input reduction* and *window extension*. The first baseline is **Vanilla**. It directly consumes tokens until the context window of LLM is fully utilized, implying a 200k window LLM if using Claude 3. The other one is Retrieval-Augmented Generation (**RAG**). We use the state-of-the-art retriever [79]. Following [81], we first segment the source text into chunks of 300 words and re-rank them using a retriever. Top-n chunks are then fed into the downstream LLM until the context window is fully utilized. GovReport dataset does not contain a query initially, we create a pseudo query "What is the summary of the whole government report?" as the query to rerank.

To evaluate the performance of CoA compared with possible multi-agent approaches, we carefully construct two approaches. Similar to CoA, we also assign each chunk c_i to W_i using similar instructions to generate CU_i. In these approaches, worker agents are parallel and independent while CoA is sequential. **Multi-Agent Voting** (**Merge**) Each agent directly generate an answer a_i according to c_i. A majority voting is applied to all answers a_i to decide the final answer. **Multi-Agent Hierarchical Structure** (**Hierarchical**). Inspired by [9], we propose a hierarchical framework, where the communication forms a tree structure between workers W_i and manager M. For each worker, it first judges whether c_i contains useful information. If true, it generates a communication unit CU_i. Then, all CU_i are sent to the manager M to come up with a final answer. Besides, we append an integer number L at the end of every approach to clearly remind the window size limit of LLM. For instance, "CoA (8K)" refers to the base LLM used in CoA with window size 8K.

4.2 Overall Results of CoA

Question Answering. Table 4 shows the results of Question Answering tasks on all three models. CoA (8k) outperforms Vanilla (8k) by a large margin on *all 8 datasets*, including 13.30% on NarrativeQA, 12.82% on MuSiQue, and 22.00% on Quality, for text-bison, text-unicorn, and gemini-ultra, respectively. Also, CoA (8k) outperforms RAG (8k) model for all 8 datasets using all three LLMs, demonstrating that CoA achieves higher performance than RAG. In other words, **using multi-agent LLMs outperforms RAG models**. It is also worth noting that for gemini-ultra, Vanilla (32k) improves the Vanilla (8k) baseline, yet it is still lower than CoA (8k). CoA also obtains compatible performance with with state-of-the-art models (Appendix E), while keeping robustness of window size (Appendix F.2).

Summarization and Code Completion. Table 4 shows the results of Summarization and Code Completion tasks. Similarly, CoA (8k) also outperforms all Vanilla (8k) and (32k) baselines on all three datasets, demonstrating the strong capability of CoA on various tasks. It is worth noting that for GovReport, RAG fails to improve the baseline with pseudo query. By contrast, CoA improves the performance significantly, showing that **CoA can also be applied in non-query tasks** (Appendix F.1).

Long Context LLMs. As Claude 3 models support 200k of tokens, we call these models long context models (LCM). Table 5 shows the performance of the LCM on two datasets. As can be seen, CoA (8k) outperforms Vanilla (200k) significantly, showing that with only an 8k context window, **CoA achieves a much higher performance than LCM with a 200k context window**. Also, CoA improves the performance with the samples that can be fed into a 200k context window (no truncation). Moreover, the improvements over the Vanilla (200k) and RAG (8k) become higher when the model

Table 5: Comparison with long context LLMs on NarrativeQA and BookSum. CoA significantly outperforms Claude 3 with 200k context limits. No Trun./Trun. indicates the source text in the sample is less/more than 200k tokens which does not need/needs truncation for vanilla (200k) baseline. Average is the mean value across all samples.

	claude-3-haiku			claude-3-sonnet			claude-3-opus		
	No Trun.	Trun.	Average	No Trun.	Trun.	Average	No Trun.	Trun.	Average
NarrativeQA									
Vanilla (200k)	8.00	2.50	7.17	5.58	2.44	5.15	7.23	2.35	6.56
RAG (8k)	5.94	4.22	5.71	9.09	5.17	8.50	6.13	4.29	5.86
CoA (8k)	**18.31**	**21.34**	**18.80**	**16.63**	**16.47**	**16.51**	**24.38**	**21.26**	**23.96**
BookSum									
Vanilla (200k)	11.98	11.70	12.04	12.17	11.90	12.10	14.11	13.67	14.00
CoA (8k)	**13.28**	**13.73**	**13.70**	**14.92**	**15.05**	**14.96**	**17.74**	**16.68**	**17.47**

Table 6: Comparison between CoA and other multi-agent frameworks. CoA with sequential agents outperforms other designs with multiple parallel agents including Merge and Hierarchical.

	HotpotQA	MuSiQue	NarrativeQA	Qasper	QuALITY	QMSum	GovReport	RepoBench-P
Vanilla (8k)	45.57	26.87	11.96	26.56	61.86	15.45	20.60	56.30
Merge (8k)	42.96	26.66	11.27	26.78	59.30	9.42	25.38	33.66
Hierarchical (8k)	50.62	29.40	17.04	31.39	64.20	15.19	16.54	27.96
CoA (8k)	**53.62**	**37.09**	**25.26**	**37.17**	**65.42**	**16.77**	**26.11**	**58.25**

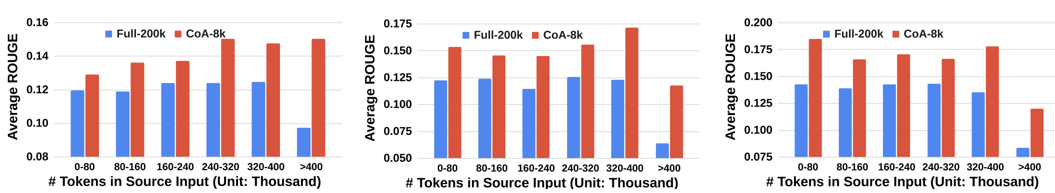
size increases from Haiku to Opus (11.63/11.36/17.4 for NarrativeQA, 1.66/2.86/3.47 for BookSum). This demonstrates that **CoA benefits from stronger models to achieve higher improvements**.

Other Multi-Agent Frameworks. As shown in Table 6, Hierarchical (8k) outperforms Vanilla (8k) on five out of eight datasets, demonstrating the hierarchical approach can also improve the vanilla baselines significantly. Merge (8k) is lower than Vanilla (8k) except for GovReport, showing that merging is especially effective for long summarization tasks such as GovReport. As can be seen, CoA outperforms Hierarchical and Merge on all eight datasets. The reason behind the results is because Hierarchical and Merge do not allow workers to communicate with each other due to their parallel designs. Thus, each worker can only maintain the information in its own chunk c_i which blocks the understanding of the whole text, hurting the performance greatly.

5 Analyses

5.1 CoA Improvement is More Obvious When RAG Fails to Retrieve Gold Answer

To demonstrate this, we first classify the samples in NarrativeQA dataset into different bins according to the position (index) of the chunk in RAG processed input that contains the gold answer. Then, we compute the average score of the CoA and RAG results of different bins. Figure 3 shows the results. As shown in the figure, RAG performs better when the index is smaller (the gold answer is nearer to the top), showing that downstream LLMs rely significantly on the quality of RAG re-ranking. Besides, the performance of RAG is positively correlated to CoA's when it successfully retrieves the gold answer. However, when RAG fails, CoA can greatly improve the performance (much higher than the tendency line).



(a) Claude 3 Haiku (b) Claude 3 Sonnet (c) Claude 3 Opus

Figure 2: Performance of Claude 3 on BookSum. Improvement is more obvious for longer inputs.

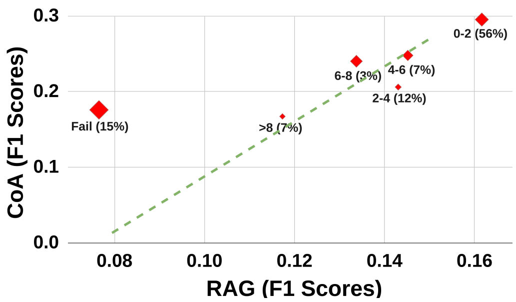


Figure 3: Comparison on NarrativeQA. X-axis/Y-axis indicate RAG/CoA performance while each point represents a bin. The number indicates the chunk index of gold answer (ratio of number of samples in bracket), and the size of the point indicates the improvement of CoA over RAG.

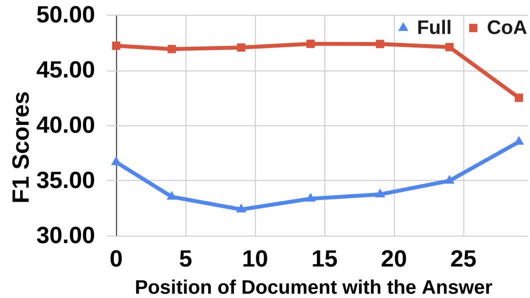


Figure 4: Performance of CoA and Full on Natural Questions. CoA mitigates the lost-in-the-middle issue. X-axis is the index of document with gold answer where small number indicates gold answer is closer to start.

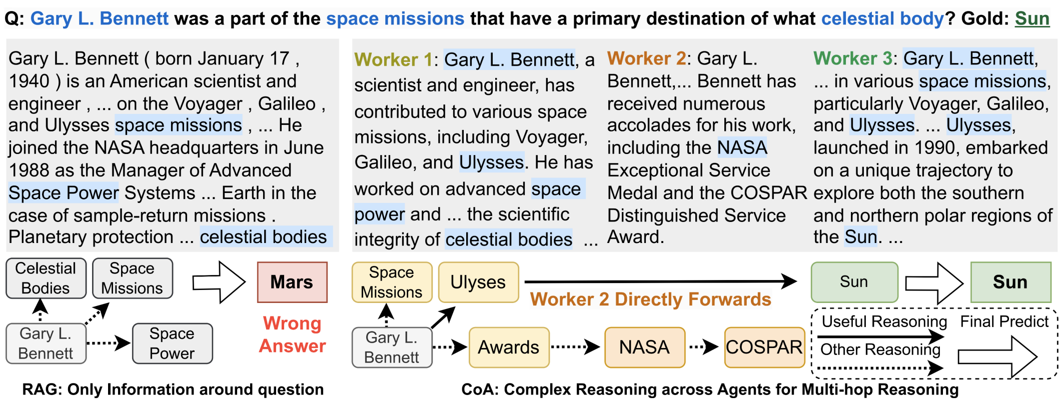


Figure 5: A case study of RAG (left) and CoA (right) on HotpotQA. The sequential agent communication enables CoA to perform complex multi-hop reasoning over long contexts.

5.2 CoA Improvement is More Obvious When Long Context Models Meet Longer Inputs

We compare the performance of CoA and Vanilla with Claude 3 on BookSum. As shown in Figure 2, CoA can outperform the vanilla baseline by a large margin on various source lengths. It is worth noting that, when the length of the sample increases, the performance even increases for CoA, and the improvement over Vanilla (200k) baseline becomes more significant. The improvement of CoA reaches around 100% when the length is larger than 400k. Thus, we can conclude that 1) CoA can still enhance the LLM performance even though the model has a very long context window limit; and 2) CoA delivers more performance gains when the input is longer.

5.3 CoA Mitigates "Lost-in-the-Middle" Phenomenon

To assess the "lost-in-the-middle" [42] effect on Vanilla and CoA models, we replicated the original study by randomly selecting 500 samples from their dataset to create a QA dataset. The results are displayed in Figure 4. The Vanilla model exhibits a significant "lost-in-the-middle" issue, with a performance range of 6.13 ($\pm$2.17). In contrast, CoA shows resilience against this issue, with a narrower performance gap of 4.89 ($\pm$1.91), demonstrating that CoA effectively mitigates this problem by providing each agent a shorter context to focus on.

5.4 Multi-agent Collaboration in CoA Enables Complex Reasoning over Long Context

Figure 5 displays a sample prediction from HotpotQA. To find the correct answer, RAG retrieves text chunks with high semantic similarity with the *query*. However, conducting multi-hop reasoning is challenging as the critical first-hop *answer* often lacks semantic relevance to the *query*. In contrast, CoA operates differently: the first agent explores related topics without knowing the query's answer, aiding subsequent inference. The second agent, also unaware of the answer, broadens the topic scope by incorporating new information. The third agent finally discovers the answer, synthesizing

Table 7: Ablation on CoA. Manager plays an important role in CoA, and left-to-right yields the best performance among other reading orders including Right-to-Left and Permutation.

	HotpotQA	MuSiQue	NarrativeQA	Qasper	QuALITY	QMSum	RepoBench-P
CoA	53.62	**37.09**	**25.26**	**37.17**	**65.42**	**16.77**	58.25
w/o Manager	48.58	26.79	20.80	29.66	58.80	16.50	56.16
Right-to-Left	51.83	29.77	21.57	36.60	62.80	15.91	55.10
Permutation	**56.05**	34.55	23.60	37.42	64.60	16.50	**58.43**

Table 8: Comparison of three multi-path augmentation through judge or voting. Multi-path CoA furthers enhance the performance.

	HotpotQA	MuSiQue	NarrativeQA	Qasper	QuALITY	QMSum	RepoBench-P
Bi-direction: left-to-right and right-to-left paths (2-way)							
w/ judge	54.11	36.97	24.47	39.23	65.00	16.41	49.69
w/ vote	51.46	34.16	21.59	37.42	64.60	9.51	38.70
oracle	62.12	48.02	32.84	46.37	71.80	18.83	59.67
Self-Consistency: five left-to-right reasoning paths (5-way)							
w/ judge	57.17	38.82	21.58	36.24	62.80	17.06	46.97
w/ vote	57.49	40.78	25.56	39.15	68.60	8.35	35.56
oracle	67.07	55.74	39.89	52.74	80.40	20.81	63.52
Permutation: five random order reasoning paths (5-way)							
w/ judge	59.17	42.37	25.47	37.65	63.40	17.81	52.45
w/ vote	58.29	39.17	26.58	38.09	67.60	8.31	35.44
oracle	75.73	60.16	39.58	52.22	79.80	20.88	67.80

information from earlier agents and new data to complete the reasoning chain. This collaborative approach highlights CoA's ability to facilitate complex reasoning across long context tasks.

5.5 Ablation Study: Effectiveness of Manager and Alternative Design Choices

To demonstrate the effect of the manager, we conduct an ablation study that uses the last worker to generate results directly. As shown in Table 7, "w/o Manager" hurts the performance significantly, dropping more than 10% on MuSiQue. This demonstrates the important role of the manager. Next, to empirically verify that left-to-right yields the best performance, we evaluate other orders of reading, including Right-to-Left by reading from the last chunk to the first one and Permutation which reads in random order. As shown in Table 7, on most of the datasets, left-to-right yields the highest score, demonstrating the advantages of natural reading order.

5.6 Multi-path Chain-of-Agents Further Enhances Performance

We manually investigated the results over these three orders (left-to-right, right-to-left, permutation), and we found that other orders sometimes can produce better answers than left-to-right. Inspired by this observation, we explore two approaches to select the best result among multiple paths. w/ vote applies majority voting over the final results while w/ judge uses an LLM to judge the most reliable CU_l of diverse paths and generate the final answer. Oracle picks the best path by evaluating score of each path, yielding the upper bound performance. Table 8 compares three multi-path augmentation approaches. Surprisingly, results show that 1) all ensemble approaches (Bi-direction, Self-consistency, and Permutation) can further enhance the performance of CoA and 5-way Permutation yields the best improvement, 2) majority voting (w/ vote) of final answer is better than using an LLM as judge (w/ judge) in Self-consistency, but worse in Bi-direction, 3) using LLM judge (w/ judge) works well on long result generation tasks (QMSum, RepoBench-P), and 4) there is large space to improve because oracle (choose as answer the one with highest performance) is much higher than either w/ judge or w/ vote. We leave the direction of multi-path reasoning to future study.

5.7 Practical Time Complexity

We run an analysis to show the practical time consumption of the proposed CoA on HotpotQA dataset. We choose llama3-8b as the backbone model for preventing additional untrackable latency due to the

Table 9: Practical time analysis on HotpotQA dataset. Avg. # of Input/Output shows the total input and output tokens for each model.

	Running Time (s)	Avg. # of Input	Avg. # of Output	Avg. # Agent Output
Vanilla (8k)	1.33	5912.85	2.40	2.40
RAG (8k)	2.41	16479.91	2.75	2.75
CoA (8k)	3.10	10840.95	38.38	11.30

Table 10: Information loss of text-bison model on different datasets.

	HotpotQA	MuSiQue	Qasper	NarrativeQA	QMSum
CoA performance	54.87	40.38	37.03	24.04	17.01
Information Loss	1.46	1.65	3.92	3.88	0.91

network of API queries. As can be seen in the Table 9, Vanilla consumes the least amount of tokens and generates the least of tokens as well. However, it truncates the input thus maintaining a low performance. Although RAG generates fewer tokens than CoA (by adding retrieval and downstream input together), it needs to read the whole input by Retrieval models which also consumes additional time. Overall, RAG is faster than CoA by around 30% in this example. **Parallel decoding.** For decoder-only LLMs, CoA agents can run in parallel. Before CUs are produced, agents can start to read their assigned paragraphs and wait for the CUs to come. We conduct an approximation by asking the model to generate one token for each sample. The output time is shown to be short and negligible, mimicking the encoding time of each sample. We have found that the running time of CoA can be reduced by 57.21% on average, leading to a 1.32-second running time for each sample, closing to the running time of the Vanilla baseline.

5.8 Information Loss

To probe the information loss during information propagation in CU_i, we further propose a metric to analyze. For each sample, we compute the highest score between the communication unit and the gold answer, and if this score is higher than the score of the final prediction, we compute the difference and refer to this information loss, formulated as Infomation_Loss $= \max(0, \max_{i=0}^{l} \text{score}(CU_i, Y) - \text{score}(\hat{Y}, Y))$ where score(*) is the performance metric for each task and dataset, Y is the gold answer and $\hat{Y}$ is the prediction. Table 10 shows the information loss of text-bison. If choosing the communication unit with the highest performance, around 1%-4% performance gain can be obtained, meaning that 1%-4% of the information is lost during the chain communication. While every method might have information loss, CoA design ensures final results yielding negligible information loss.

6 Conclusion

In this paper, we propose Chain-of-Agents, a multi-agent LLM collaboration framework for solving long context tasks. It is a training free, task/length agnostic, interpretable, and cost-effective framework. Experiments show that Chain-of-Agents outperforms RAG and Long Context LLMs by a large margin despite of its simple design. Analysis shows that by integrating information aggregation and context reasoning, CoA mitigates lost-in-the-middle and performs better on longer samples.

Limitations. While CoA features with a simple and effective design, future directions can address the following limitations to further improve its prowess and efficiency. First, communication effectiveness can be further improved via finetuning or in-context learning because current LLMs are aligned with human norms which is not optimal for communication between LLMs. Second, CoA does not explore other forms of communication approaches, such as debating or complex discussions. Third, the cost and latency of running CoA can be further reduced, such as replacing some LLMs with more effective models via model routing [62].

Acknowledgement

We thank colleagues in Cloud AI Research team for providing helpful feedback for this paper. We also thank the anonymous reviewers for their helpful comments.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Qingyao Ai, Ting Bai, Zhao Cao, Yi Chang, Jiawei Chen, Zhumin Chen, Zhiyong Cheng, Shoubin Dong, Zhicheng Dou, Fuli Feng, et al. Information retrieval meets large language models: a strategic report from chinese ir community. *AI Open*, 4:80–90, 2023.

[3] Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, and Jian-Guang Lou. Make your llm fully utilize the context. *arXiv preprint arXiv:2404.16811*, 2024.

[4] Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

[5] AI Anthropic. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 2024.

[6] 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. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*, 2023.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[9] Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. Walking down the memory maze: Beyond context limit through interactive reading. *arXiv preprint arXiv:2310.05029*, 2023.

[10] Huaben Chen, Wenkang Ji, Lufeng Xu, and Shiyu Zhao. Multi-agent consensus seeking via large language models. *arXiv preprint arXiv:2310.20151*, 2023.

[11] Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. Reconcile: Round-table conference improves reasoning via consensus among diverse llms. *arXiv preprint arXiv:2309.13007*, 2023.

[12] Mingda Chen, Zewei Chu, Sam Wiseman, and Kevin Gimpel. SummScreen: A dataset for abstractive screenplay summarization. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8602–8615, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[13] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*, 2023.

[14] Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas Roy, and Chuchu Fan. Scalable multi-robot collaboration with large language models: Centralized or decentralized systems? In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4311–4317. IEEE, 2024.

[15] Nelson Cowan. Working memory underpins cognitive development, learning, and education. *Educational psychology review*, 26:197–223, 2014.

[16] Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, Online, June 2021. Association for Computational Linguistics.

[17] Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*, 2021.

[18] Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

[19] Navid Ghaffarzadegan, Aritra Majumdar, Ross Williams, and Niyousha Hosseinichimeh. Generative agent-based modeling: Unveiling social system dynamics through coupling mechanistic models with generative artificial intelligence. *arXiv preprint arXiv:2309.11456*, 2023.

[20] Daya Guo, Canwen Xu, Nan Duan, Jian Yin, and Julian McAuley. Longcoder: A long-range pre-trained language model for code completion. In *International Conference on Machine Learning*, pages 12098–12107. PMLR, 2023.

[21] Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024.

[22] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In Donia Scott, Nuria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.

[23] Wenyue Hua, Lizhou Fan, Lingyao Li, Kai Mei, Jianchao Ji, Yingqiang Ge, Libby Hemphill, and Yongfeng Zhang. War and peace (waragent): Large language model-based multi-agent simulation of world wars. *arXiv preprint arXiv:2311.17227*, 2023.

[24] Dong Huang, Qingwen Bu, Jie M Zhang, Michael Luck, and Heming Cui. Agentcoder: Multi-agent-based code generation with iterative testing and optimisation. *arXiv preprint arXiv:2312.13010*, 2023.

[25] Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. Efficient attentions for long document summarization. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436, Online, June 2021. Association for Computational Linguistics.

[26] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43, 2023.

[27] Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *arXiv preprint arXiv:2310.06839*, 2023.

[28] Zhengbao Jiang, Luyu Gao, Zhiruo Wang, Jun Araki, Haibo Ding, Jamie Callan, and Graham Neubig. Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2336–2349, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

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

[30] Tushar Khot, H. Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. *ArXiv*, abs/2210.02406, 2022.

[31] Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018.

[32] Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The NarrativeQA Reading Comprehension Challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 05 2018.

[33] Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. Booksum: A collection of datasets for long-form narrative summarization. *arXiv preprint arXiv:2105.08209*, 2021.

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

[35] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems*, 36:51991–52008, 2023.

[36] Nian Li, Chen Gao, Yong Li, and Qingmin Liao. Large language model-empowered agents for simulating macroeconomic activities. *arXiv preprint arXiv:2310.10436*, 2023.

[37] Tianle Li, Ge Zhang, Quy Duc Do, Xiang Yue, and Wenhu Chen. Long-context llms struggle with long in-context learning. *arXiv preprint arXiv:2404.02060*, 2024.

[38] Yucheng Li. Unlocking context constraints of llms: Enhancing context efficiency of llms with self-information-based content filtering. *arXiv preprint arXiv:2304.12102*, 2023.

[39] Jonathan Light, Min Cai, Sheng Shen, and Ziniu Hu. Avalonbench: Evaluating llms playing the game of avalon. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.

[40] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, 2004. Association for Computational Linguistics.

[41] Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6385–6400, Singapore, December 2023. Association for Computational Linguistics.

[42] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

[43] Tianyang Liu, Canwen Xu, and Julian McAuley. Repobench: Benchmarking repository-level code auto-completion systems. *arXiv preprint arXiv:2306.03091*, 2023.

[44] Xuezhe Ma, Xiaomeng Yang, Wenhan Xiong, Beidi Chen, Lili Yu, Hao Zhang, Jonathan May, Luke Zettlemoyer, Omer Levy, and Chunting Zhou. Megalodon: Efficient llm pretraining and inference with unlimited context length. *arXiv preprint arXiv:2404.08801*, 2024.

[45] Zhao Mandi, Shreeya Jain, and Shuran Song. Roco: Dialectic multi-robot collaboration with large language models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 286–299. IEEE, 2024.

[46] Amirkeivan Mohtashami and Martin Jaggi. Landmark attention: Random-access infinite context length for transformers. *arXiv preprint arXiv:2305.16300*, 2023.

[47] Gabriel Mukobi, Hannah Erlebach, Niklas Lauffer, Lewis Hammond, Alan Chan, and Jesse Clifton. Welfare diplomacy: Benchmarking language model cooperation. *arXiv preprint arXiv:2310.08901*, 2023.

[48] Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. Leave no context behind: Efficient infinite context transformers with infini-attention. *arXiv preprint arXiv:2404.07143*, 2024.

[49] Bo Ni and Markus J Buehler. Mechagents: Large language model multi-agent collaborations can solve mechanics problems, generate new data, and integrate knowledge. *Extreme Mechanics Letters*, page 102131, 2024.

[50] OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023.

[51] Richard Yuanzhe Pang, Alicia Parrish, Nitish Joshi, Nikita Nangia, Jason Phang, Angelica Chen, Vishakh Padmakumar, Johnny Ma, Jana Thompson, He He, and Samuel Bowman. QuALITY: Question answering with long input texts, yes! In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5336–5358, Seattle, United States, July 2022. Association for Computational Linguistics.

[52] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22, 2023.

[53] Joon Sung Park, Lindsay Popowski, Carrie Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Social simulacra: Creating populated prototypes for social computing systems. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, pages 1–18, 2022.

[54] Ethan Perez, Patrick Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. Unsupervised question decomposition for question answering. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8864–8880, Online, November 2020. Association for Computational Linguistics.

[55] Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

[56] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019.

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

[58] Swarnadeep Saha, Omer Levy, Asli Celikyilmaz, Mohit Bansal, Jason Weston, and Xian Li. Branch-solve-merge improves large language model evaluation and generation. *arXiv preprint arXiv:2310.15123*, 2023.

[59] Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. ZeroSCROLLS: A zero-shot benchmark for long text understanding. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7977–7989, Singapore, December 2023. Association for Computational Linguistics.

[60] Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, and Omer Levy. SCROLLS: Standardized CompaRison over long language sequences. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 12007–12021, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[61] Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*, 2023.

[62] Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. Large language model routing with benchmark datasets. *arXiv preprint arXiv:2309.15789*, 2023.

[63] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*, 2022.

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

[65] Simeng Sun, Y. Liu, Shuo Wang, Chenguang Zhu, and Mohit Iyyer. Pearl: Prompting large language models to plan and execute actions over long documents. *ArXiv*, abs/2305.14564, 2023.

[66] Xiangru Tang, Anni Zou, Zhuosheng Zhang, Yilun Zhao, Xingyao Zhang, Arman Cohan, and Mark Gerstein. Medagents: Large language models as collaborators for zero-shot medical reasoning. *arXiv preprint arXiv:2311.10537*, 2023.

[67] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[68] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[69] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

[70] Bing Wang, Changyu Ren, Jian Yang, Xinnian Liang, Jiaqi Bai, Qian-Wen Zhang, Zhao Yan, and Zhoujun Li. Mac-sql: Multi-agent collaboration for text-to-sql. *arXiv preprint arXiv:2312.11242*, 2023.

[71] Boshi Wang, Xiang Deng, and Huan Sun. Iteratively prompt pre-trained language models for chain of thought. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2714–2730, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[72] Boxin Wang, Wei Ping, Peng Xu, Lawrence McAfee, Zihan Liu, Mohammad Shoeybi, Yi Dong, Oleksii Kuchaiev, Bo Li, Chaowei Xiao, Anima Anandkumar, and Bryan Catanzaro. Shall we pretrain autoregressive language models with retrieval? a comprehensive study. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7763–7786, Singapore, December 2023. Association for Computational Linguistics.

[73] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

[74] Shenzhi Wang, Chang Liu, Zilong Zheng, Siyuan Qi, Shuo Chen, Qisen Yang, Andrew Zhao, Chaofei Wang, Shiji Song, and Gao Huang. Avalon's game of thoughts: Battle against deception through recursive contemplation. *arXiv preprint arXiv:2310.01320*, 2023.

[75] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Huai hsin Chi, F. Xia, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903, 2022.

[76] Martin Weiss, Nasim Rahaman, Manuel Wuthrich, Yoshua Bengio, Li Erran Li, Bernhard Schölkopf, and Christopher Pal. Rethinking the buyer's inspection paradox in information markets with language agents, 2023.

[77] Tongshuang Wu, Michael Terry, and Carrie Jun Cai. Ai chains: Transparent and controllable human-ai interaction by chaining large language model prompts. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery.

[78] Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighof. C-pack: Packaged resources to advance general chinese embedding. *arXiv preprint arXiv:2309.07597*, 2023.

[79] Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. C-pack: Packaged resources to advance general chinese embedding, 2023.

[80] Kai Xiong, Xiao Ding, Yixin Cao, Ting Liu, and Bing Qin. Examining inter-consistency of large language models collaboration: An in-depth analysis via debate. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

[81] Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. Retrieval meets long context large language models. *arXiv preprint arXiv:2310.03025*, 2023.

[82] Yuzhuang Xu, Shuo Wang, Peng Li, Fuwen Luo, Xiaolong Wang, Weidong Liu, and Yang Liu. Exploring large language models for communication games: An empirical study on werewolf. *arXiv preprint arXiv:2309.04658*, 2023.

[83] Zelai Xu, Chao Yu, Fei Fang, Yu Wang, and Yi Wu. Language agents with reinforcement learning for strategic play in the werewolf game. *arXiv preprint arXiv:2310.18940*, 2023.

[84] Zhenran Xu, Senbao Shi, Baotian Hu, Jindi Yu, Dongfang Li, Min Zhang, and Yuxiang Wu. Towards reasoning in large language models via multi-agent peer review collaboration. *arXiv preprint arXiv:2311.08152*, 2023.

[85] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[86] Hongbin Ye, Honghao Gui, Aijia Zhang, Tong Liu, Wei Hua, and Weiqiang Jia. Beyond isolation: Multi-agent synergy for improving knowledge graph construction. *arXiv preprint arXiv:2312.03022*, 2023.

[87] Dani Yogatama, Cyprien de Masson d'Autume, and Lingpeng Kong. Adaptive semiparametric language models. *Transactions of the Association for Computational Linguistics*, 9:362–373, 2021.

[88] An Zhang, Leheng Sheng, Yuxin Chen, Hao Li, Yang Deng, Xiang Wang, and Tat-Seng Chua. On generative agents in recommendation. *arXiv preprint arXiv:2310.10108*, 2023.

[89] Hongxin Zhang, Weihua Du, Jiaming Shan, Qinhong Zhou, Yilun Du, Joshua B Tenenbaum, Tianmin Shu, and Chuang Gan. Building cooperative embodied agents modularly with large language models. *arXiv preprint arXiv:2307.02485*, 2023.

[90] Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed H Awadallah, Dragomir Radev, and Rui Zhang. Summ^n: A multi-stage summarization framework for long input dialogues and documents. *arXiv preprint arXiv:2110.10150*, 2021.

[91] Yusen Zhang, Ansong Ni, Tao Yu, Rui Zhang, Chenguang Zhu, Budhaditya Deb, Asli Celikyilmaz, Ahmed Hassan Awadallah, and Dragomir Radev. An exploratory study on long dialogue summarization: What works and what's next. *arXiv preprint arXiv:2109.04609*, 2021.

[92] Jun Zhao, Can Zu, Hao Xu, Yi Lu, Wei He, Yiwen Ding, Tao Gui, Qi Zhang, and Xuanjing Huang. Longagent: Scaling language models to 128k context through multi-agent collaboration. *arXiv preprint arXiv:2402.11550*, 2024.

[93] Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. QMSum: A new benchmark for query-based multi-domain meeting summarization. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921, Online, June 2021. Association for Computational Linguistics.

[94] Denny Zhou, Nathanael Scharli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Huai hsin Chi. Least-to-most prompting enables complex reasoning in large language models. *ArXiv*, abs/2205.10625, 2022.

[95] Wangchunshu Zhou, Yuchen Eleanor Jiang, Peng Cui, Tiannan Wang, Zhenxin Xiao, Yifan Hou, Ryan Cotterell, and Mrinmaya Sachan. Recurrentgpt: Interactive generation of (arbitrarily) long text. *arXiv preprint arXiv:2305.13304*, 2023.

A Proof of Time Complexity

Assuming the source text containing n tokens, window limit of LLM is k tokens, and the responses contain r tokens in average. For decoder-only LLM, we grasp the operations for attention calculation as the time cost unit. Then, for Full-Context LLM, total operation for encoding input source text T_{Full} is:

$$T_{Enc} = (1 + 2 + \cdots + n) = \frac{(n+1)n}{2} = \mathcal{O}(n^2) \tag{3}$$

Similarly, decoding starts when the model already generate all input. Thus, the first decoded token attends to n positions. Total operation for decoding response is ($r \ll n$):

$$T_{Dec} = (n + 1 + n + 2 + \cdots + n + r) = \frac{(n + 1 + n + r)r}{2} = \mathcal{O}(nr + r^2) = \mathcal{O}(nr) \tag{4}$$

For Chain-of-Agents, we first split the source into $\lceil n/k \rceil$ chunks. Thus, total encoding time for all input is:

$$T_{Enc} = (1 + 2 + \cdots + k) \times \lceil n/k \rceil = \frac{(k+1)k \times \lceil n/k \rceil}{2} = \mathcal{O}(k^2 \times n/k) = \mathcal{O}(nk) \tag{5}$$

Decoding starts when the model already generates k tokens. Thus, the first decoded token attends to k positions. The total operation for decoding response is ($r \ll k$):

$$T_{Dec} = (k{+}1{+}k{+}2{+}\cdots{+}k{+}r) \times \lceil n/k \rceil = \frac{(k + 1 + k + r)r \times \lceil n/k \rceil}{2} = \mathcal{O}(nr{+}nr^2/k) = \mathcal{O}(nr) \tag{6}$$

Similarly, for RAG models, encoding time is retrieval time $\mathcal{O}(k'^2 \times n/k') = \mathcal{O}(nk')$ plus the encoding time of downstream LLM $\mathcal{O}(k^2)$. Decoding cost is the time of retrieval embedding generation $\mathcal{O}(n/k')$, plus the generation of the downstream LLM $\mathcal{O}(kr)$.

B Implementation Details

For all experiments, we use Vertex model garden [4] API to use all six models. Maximum generation token is set to 2048 for gemini-ultra and set to 1024 for the rest of the models. We set temperature to 0 for all experiments except for Self-consistency setting. Table 11 shows the prompt for all models. for task specific requirement of 9 datasets, we follow the original LongBench [6] and SCROLLS [60]. For RAG model, we use the model provided by Huggingface[5] and run on A100 GPUs to rerank the chunks. Algorithm 2 shows the algorithm to chunk the input to cater the window size.

Algorithm 2 Chain of Agents (CoA) Input Chunking Algorithm.

Input: Source input x, query q, agent window size k, instruction I_w.
Output: A list of chunks $C = \{c_1, c_2, \cdots, c_l\}$.
  Split the source x into sentences $s_1, s_2, \cdots, s_n$
  Initialize $c \leftarrow$ empty string, $C \leftarrow$ empty ordered set,
  Initialize length budgets $B \leftarrow k -$ count_token$(q) -$ count_token(I_w).
  for s in $s_1, s_2, \cdots, s_n$ **do**
    if count_token(c) + count_token(s) > B **then**
      Append c to C, initialize $c \leftarrow$ empty string
    end if
    $c \leftarrow c \bigoplus s$ // $\bigoplus$ indicates concatenating two strings with a blank.
  end for
  if count_token(c) $\neq 0$ **then**
    Append c to C
  end if
  return C

[4] https://cloud.google.com/model-garden

[5] https://huggingface.co/

Table 11: Prompt of all models for query-based tasks.

Vanilla	{Task specific requirement} {Source Input x with truncation if needed} Question: {Question q} Answer:
RAG	{Task specific requirement} {Retrieved Chunks of Source Input x} Question: {Question q} Answer:
CoA	**Worker** W_i**:** {Input Chunk c_i} Here is the summary of the previous source text: {Previous Communication Unit (CU_{i-1})} Question: {Query q} You need to read current source text and summary of previous source text (if any) and generate a summary to include them both. Later, this summary will be used for other agents to answer the Query, if any. So please write the summary that can include the evidence for answering the Query: **Manager** M**:** {Task specific requirement} The following are given passages. However, the source text is too long and has been summarized. You need to answer based on the summary: {Previous Communication Unit CU_l} Question: {question} Answer:

Table 12: Prompt of all models for non-query-based tasks.

Vanilla	{Task specific requirement} {Source Input x with truncation if needed} Answer:
RAG	{Task specific requirement} {Retrieved Chunks of Source Input x} Answer:
CoA	**Worker** W_i**:** {Input Chunk c_i} Here is the summary of the previous source text: {Previous Communication Unit (CU_{i-1})} You need to read the current source text and summary of previous source text (if any) and generate a summary to include them both. Later, this summary will be used for other agents to generate a summary for the whole text. Thus, your generated summary should be relatively long. **Manager** M**:** {Task specific requirement} The following are given passages. However, the source text is too long and has been summarized. You need to answer based on the summary: {Previous Communication Unit CU_l} Answer:

C Example Outputs

We pick one sample from HotpotQA in Table 13, QMSum in Table 14, and RepoBench-P in Table 15 to show the prediction of Chain-of-Agents, in the order of question q, Communication Unit CU of agents, and final prediction of manager. We also observe that 1) the first worker usually explores the most number of paths, including 3 or more topics because the agent is not sure about the result and willing to provide more information for the next ones. 2) The final worker usually narrows down to one answer with a much shorter CU, and 3) The speed of narrowing down will be faster for simpler samples, producing a small reasoning graph.

Table 13: Example output on HotpotQA dataset.

Question	Gary L. Bennett was a part of the space missions that have a primary destination of what celestial body?
Worker 1	Gary L. Bennett, a scientist and engineer, has contributed to various space missions, including Voyager, Galileo, and Ulysses. He has worked on advanced space power and propulsion systems and has been involved in planetary protection measures. Bennett's expertise has been instrumental in ensuring the scientific integrity of celestial bodies and preventing harmful contamination. He has received numerous awards and accolades for his contributions to space exploration and is recognized as a leading expert in the field of planetary protection.
Worker 2	Gary L. Bennett, a renowned scientist and engineer, has made significant contributions to space missions, including Voyager, Galileo, and Ulysses. His expertise in advanced space power and propulsion systems, as well as planetary protection measures, has been crucial in ensuring the scientific integrity of celestial bodies and preventing harmful contamination. Bennett has received numerous accolades for his work, including the NASA Exceptional Service Medal and the COSPAR Distinguished Service Award.
Worker 3	Gary L. Bennett, a distinguished scientist and engineer, played a pivotal role in various space missions, particularly Voyager, Galileo, and Ulysses. His expertise in advanced space power and propulsion systems, coupled with his focus on planetary protection measures, has been instrumental in safeguarding the scientific integrity of celestial bodies. Ulysses, launched in 1990, embarked on a unique trajectory to explore both the southern and northern polar regions of the Sun. During its extended mission, Ulysses provided invaluable data on the Sun's magnetic field, solar wind, and the presence of dust in the Solar System.
Manager	Sun
RAG	Mars
Gold	Sun

Table 14: Example output on QMSum dataset.

Question	Summarize the discussion about industrial components.
Worker 1	The industrial designer proposed that the remote control should be made of titanium or rubber, and the buttons should be simple. The marketing expert said that the remote control should be fancy, innovative, and easy to use. They also discussed the trend of making products look like fruit and vegetables.
Worker 2	The industrial designer proposed that the remote control should be made of titanium or rubber, and the buttons should be simple. The marketing expert said that the remote control should be fancy, innovative, and easy to use. They also discussed the trend of making products look like fruit and vegetables. The industrial designer said that they would explore the two options of titanium and rubber, and see if rubber is expensive. The user interface designer said that they would pretend that the modelling clay is titanium and paint it afterwards. The project manager said that they would stick with two batteries and not reinvent the wheel.
Manager	The industrial designer proposed that the remote control should be made of titanium or rubber, and the buttons should be simple. The marketing expert said that the remote control should be fancy, innovative, and easy to use. They also discussed the trend of making products look like fruit and vegetables. The industrial designer said that they would explore the two options of titanium and rubber, and see if rubber is expensive. The user interface designer said that they would pretend that the modelling clay is titanium and paint it afterwards. The project manager said that they would stick with two batteries and not reinvent the wheel. (ROUGE Score: 21.38)
RAG	The industrial designer and the marketing manager discussed the industrial components of the remote control. They agreed that the remote control should have a simple design, with a few buttons and a speech recognition technology. They also discussed the possibility of adding a L_C_D_ display, but decided that it would be too expensive. The designer suggested that the remote control could be made of titanium or rubber, and the marketing manager agreed that a spongy design would be more appealing to consumers. (ROUGE Score: 16.54)
Gold	The industrial designer provided several options respectively for energy, material and interface. Among these options, the industrial designer preferred traditional battery, titanium and simple push buttons so that they would have enough money for speech recognition. After that, the user interface designer proposed an LCD display only for output, which might be taken into account. The group also talked about how to enable the users to find their controllers.

Table 15: Example output on RepoBench-P dataset.

Question	What is the next line of code: ... if(type == 5 || type == 0) {
Worker 1	The method SensorDataCollectorManager.flushSensorDataCache(int type, String deviceID) flushes the database cache for the given sensor type. If the type is 0, all sensor types are flushed. The method first checks if the type is valid, and if it is, it calls the flushDBCache method for the appropriate sensor collector. SensorDataCollectorManager.flushSensorDataCache(int type, String deviceID) flushes the database cache for the given sensor type. If the type is 0, all sensor types are flushed. The method first checks if the type is valid, and if it is, it calls the flushDBCache method for the appropriate sensor collector.
Worker 2	The method SensorDataCollectorManager.flushSensorDataCache(int type, String deviceID) flushes the database cache for the given sensor type. If the type is 0, all sensor types are flushed. The method first checks if the type is valid, and if it is, it calls the flushDBCache method for the appropriate sensor collector.
Manager	LightSensorCollector.flushDBCache(deviceID);
RAG	AccelerometerSensorCollector.flushDBCache(deviceID);
Gold	LightSensorCollector.flushDBCache(deviceID);

Table 16: Compare between CoA and previous state-of-the-art models on nine datasets. * indicates the model needs further training.

	HotpotQA	MuSique	Qasper	NarrativeQA	Quality	QMSum	GovReport	BookSum	RepoBench-P
Previous Best	54.4 [29]	40.4 [29]	**53.9*** [60]	**26** [29]	**89.2** [59]	**22.44*** [60]	26.3 [60]	**18.5*** [78]	56.47 [29]
Ours Best	**62.04**	**42.49**	38.01	25.26	83.8	17.67	**26.98**	17.47	**73.05**

D Broader Impacts

Chain-of-Agents is a generic framework for long context tasks. users can apply this to diverse tasks not restricting to the mentioned ones. It will greatly increase the efficiency of individuals or companies to solve complex long context tasks. Besides, the interpretablity of such approach can reduce the misuse of the LLMs because users can check the correctness of results and decrease the possibility of making faults. However, similar to all prompt based approaches, this framework requires careful prompt design for unseen large language models, users may not get optimal solution on certain newly proposed LLMs. Besides, it may increase the number of the calls for API, causing higher network traffic and higher latency for user pools.

E Comparison with State-of-the-Art Models

Table 16 shows the comparisons of CoA with the previous state-of-the-art performance, including the ones requiring training (indicated with *). As can be seen, CoA achieves better or comparable results on all datasets, improving HotpotQA and RepoBench-p by a large margin. The performance on some datasets is lower than that of state-of-the-art models because the datasets are trained on some domain-specific models, such as Qasper.

F More Experiments and Analysis

F.1 Short LLMs on BookSum Dataset

We have run the BookSum dataset with Text-bison and Text-unicorn models. As shown in the Table 17, similar to the results with the LLMs that support longer contexts, CoA outperforms baselines by a significant margin.

F.2 Robustness against Context Window Size

We set the default context window of CoA to 8k due to the limitation of text-bison and unicorn models. To test the influence of CoA against context window change, we set window sizes to 4k, 8k,

Table 17: Performance of text-bison and text-unicorn on BookSum dataset.

	text-bison	text-unicorn
Vanilla (8k)	9.13	8.15
RAG (8k)	9.38%	8.01
CoA (8k)	14.51	14.41

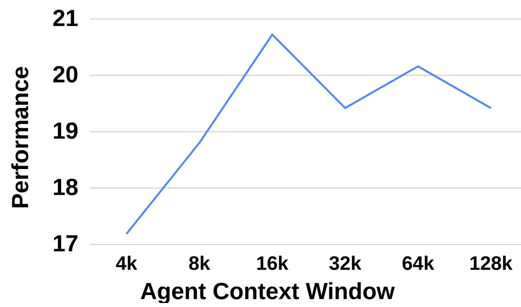


Figure 6: Performance of CoA on Claude 3 Haiku on the NarrativeQA dataset with various context window sizes of an agent. Results show the robustness of CoA towards different choices of context lengths.

16k, 32k, 64k, and 128k of Claude 3 Haiku model and evaluate on NarrativeQA dataset and see the performance change. As shown in Figure 6, the performance of the model increases from 4k to 16k and stabilizes to around 20 with the context window further increases. This result shows that CoA will benefit from increasing length and keep stable when the length touches a bound.

We further analyze different agent lengths and reported the scores of text-bison-32k in Table 18. Our results show that it can improve the baseline with various window sizes. Thus, we choose 8k for short context models, such as text-bison.

F.3 Evaluation on NIAH Test

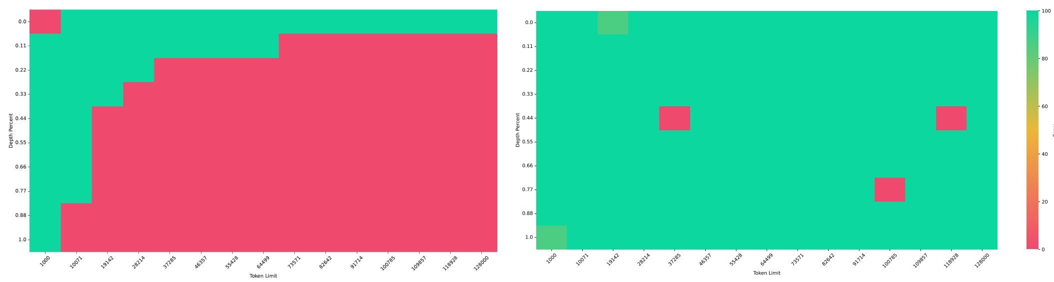
We conduct evaluations on the NIAH test. We follow the LongAgent [92] paper to run a NIAH PLUS test for evaluating the long context understanding capability of LLMs. Different from the original NeedleInAHaystack test, NIAH PLUS is more challenging because LLMs need to answer the questions rather than simply retrieve the information (needle). The results are shown in Figure 7. As can be seen, the CoA greatly increases the accuracy of Text-bison from 26.0% to 97.8%, showing that CoA significantly increases the capability of LLMs to understand long contexts.

F.4 Preventing Collapse of Reasoning Chain

Experiments show that when certain irrelevant content is present in the communication unit CU_{i-1}, it often prevents the worker unit W_i from generating a meaningful response. This meaningless response then propagates to the next unit, causing W_{i+1} to produce similarly meaningless content and eventually leading to a collapse of the entire reasoning chain. We call this phenomenon catastrophic collapse. In such cases, the model's overall performance may even fall below that of a single agent W_1. A common instance of this phenomenon is a rejection response. If CU_{i-1} includes rejection cues such as "I don't know the answer to the query" or "not mentioned", W_i is likely to repeat

Table 18: Comparison between vanilla and CoA with various context lengths using text-bison-32k.

	4k	8k	16k	32k
text-bison-32k	45.69	53.55	59.14	48.54
Ours (same base)	**54.95**	**60.34**	**63.11**	**50.25**



(a) Vanilla (8k) (b) CoA (8k)

Figure 7: NIAH PLUS test results of Vanilla model and CoA model. Greener/Redder cells indicate higher/lower performance. CoA with 8k improves the performance of the NIAH test greatly compared with the Vanilla baseline.

Table 19: Comparison of related work and their topologies.

Name or Source	Task or Challenge	Agent Structure	Communication Schema
[14]	Multi-robot planning	Decentralized	Circle
RoCo [45]	Multi-robot collaboration	Decentralized	Planning Pipeline
CoELA [89]	Multi-Agents cooperation	Decentralized	Memory Module
MAD [18]	Improving Factuality	Decentralized	Debate
Reconclie [11]	Reasoning	Decentralized	Round Table
WalkMaze [9]	Long Input	Centralized	Tree Structure
LongAgent [92]	Long Input	Centralized	Tree Structure
RecurentGPT [95]	Long Output	Decentralized	Memory Gating
CoA (Ours)	Long Input	Decentralized	Chain of Agents

similar content without adding any meaningful information, often leading to a final output of "not mentioned".

To mitigate this issue, three simple methods can be used: (1) repeatedly running the sample until the final result is not a rejection; (2) postprocessing the communication unit to remove irrelevant words; and (3) using prompt engineering to instruct worker agents to ignore irrelevant information in CU_i. We speculate that this issue may be influenced by reinforcement learning from human feedback (RLHF) or chat fine-tuning, causing verbose responses that introduce excessive noise. Further fine-tuning may be needed to address this problem, which we leave to future work.

G Detailed Comparison with Related Works

Table 19 illustrates the comparison of related work on multiagent LLM systems. Broadly, prior work on multi-agent LLM systems for long context tasks has a centralized design, where the sibling worker agents do not communicate during inference the communication unit only propagates between manager and worker, and existing decentralized designs do not extend effectively to long context tasks. To the best of our knowledge, CoA is the first to use a decentralized structure on long input tasks and is more effective than baselines such as RAG.

The challenge of long dependency on input remains under-explored (as illustrated in Table 20). Our target is to novelly mitigate challenging context dependency issues in long-context tasks particularly those requiring complex reasoning. To this end, we explored various structures (WalkMaze, Merge, Debate, Group Discussion, etc.) and found that a variant of decentralized well-existing chain communication and this simple approach can work more effectively than others. Our contribution is to mitigate a challenging issue (long input dependency) with a simple intuitive approach (decentralized chain communication). We want to emphasize that centralized communication and our work solve the long context problems in different ways, and they are not in conflict with each other. They can be merged into a much stronger framework (e.g., adding our chain reference stage to mitigate context dependency issues before the stages in LongAgent [92]).

Table 20: Example for comparison of Centralized and Decentralized models.

Input Source	**Question**: Who is the grandson of A? **Source**: [1],[2],[3],[4] (chunks) **Evidence in each chunk**: [1: A's husband is D], [2: A's son is B], [3: No evidence], [4: B's son is C]
Centralized [92]	**Round 1**: Manager: Who is the son of A? Worker with [2]: It is B. Others say unknown **Round 2**: Manager: Who is the son of B? Worker with [4]: It is C. Others say unknown Final answer: It is C.
Decentralized (ours)	Manager: Who is the grandson of A? **Worker with [1]**: A's husband is D (topic exploration), **Worker with [2]**: A's son is B (answer first hop), **Worker with [3]**: A's son is B (forward previous evidence), **Worker with [4]**: A's son is B, B's son is C. Thus, A's grandson is C. (complete reasoning) **Final answer**: It is C.

Table 20 showcases the comparison between CoA and centralized designs in mitigating agent dependency issues for long inputs. Specifically, previous work for long input tasks, despite showing significant improvements, usually uses disentangling algorithms such as question decomposition, to address long-distance dependency in long inputs, and then they use centralized structure to answer the disentangled questions one by one. Different from them, CoA does not disentangle the question but leaves the entire question to the agents. Although chain communication is well-introduced in the literature, to the best of our knowledge, CoA is the first work to apply decentralized communication to long-term dependency challenges. In addition, CoA is not in conflict with existing work as it can be considered as a plugin for other multi-agent LLM systems to improve them.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our main claim that CoA improves LLM performance on long-context tasks accurately reflects our contribution and is supported by experiments.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We mention that the performance of CoA drops when the answer is in the end of long input context. Also, we show that performance of CoA can be further enhanced via multi-path ensemble and the space for improving is significant.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.

- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: Our paper does not include theoretical results.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We display all needed information in the appendix, including hyper-parameters, templates, example output, and details. These ensure reproducing the main results is easy.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.

- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will provide open access to the data and code upon acceptance.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We list the dataset details in the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We fix the temperature of the large language models. Thus, the running results are fixed. For RAG model, we also fix the random seed so that the results can be reproduced.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We conduct time complexity anlaysis to show the computer resources needed. Also, we state each of the API call clearly so that the resource information can be easily accessed.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We conform with the NeurIPS Code of Ethics including preserving anonymity.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We add a section in appendix to discus all potential social impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not release data or models.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets in our experiments are public and properly credited with license properly respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our research does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our research does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.