Are Long-LLMs A Necessity For Long-Context Tasks?

Hongjin Qian[1,2], **Zheng Liu**[2]*, **Peitian Zhang**[1], **Kelong Mao**[1], **Yujia Zhou**[1]
Xu Chen[1], **Zhicheng Dou**[1]
[1] Gaoling School of Artificial Intelligence, Renmin University of China
[2] Beijing Academy of Artificial Intelligence
{chienqhj,zhengliu1026}@gmail.com

Abstract

The learning and deployment of long-LLMs remains a challenging problem despite recent progresses. In this work, we argue that the long-LLMs are not a necessity to solve long-context tasks, as common long-context tasks are short-context solvable, i.e. they can be solved by purely working with oracle short-contexts within the long-context tasks' inputs. On top of this argument, we propose a framework called **LC-Boost** (Long-Context Bootstrapper), which enables a short-LLM to address the long-context tasks in a bootstrapping manner. In our framework, the short-LLM prompts itself to reason for two critical decisions: 1) how to access to the appropriate part of context within the input, 2) how to make effective use of the accessed context. By adaptively accessing and utilizing the context based on the presented tasks, LC-Boost can serve as a general framework to handle diversified long-context processing problems. We comprehensively evaluate different types of tasks from popular long-context benchmarks, where LC-Boost is able to achieve a substantially improved performance with a much smaller consumption of resource.

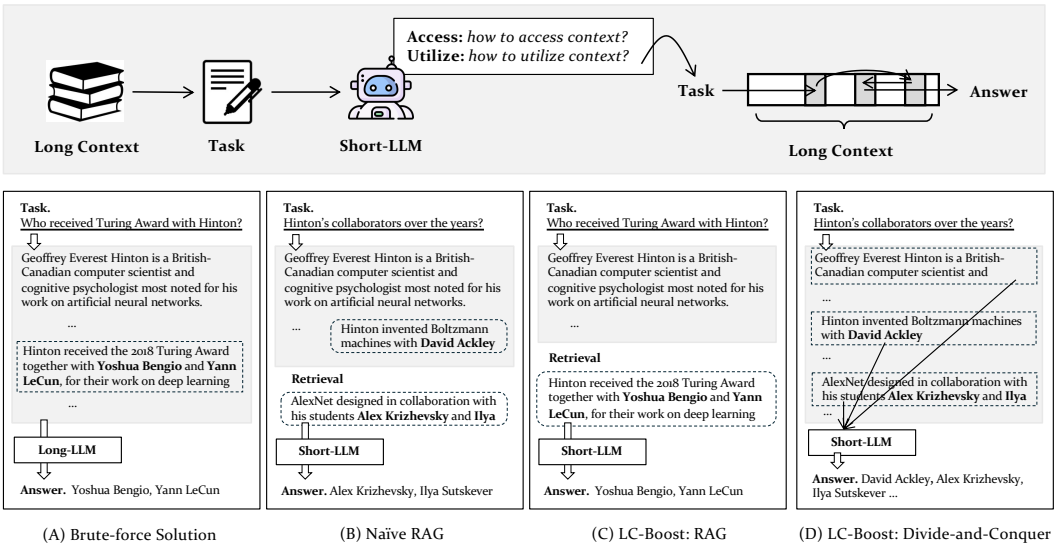
1 Introduction

Large language models (LLMs) are widely adopted for real-world applications. Many of the applications are associated with long-sequence inputs, such as long-document question answering and summarization. As such, the LLMs are commonly expected to have a long working context (*a.k.a.* long-LLMs) in order to confront such demanding scenarios [Bai et al., 2023, Zhang et al., 2024a]. Unfortunately, the learning and deployment of long-LLMs are still challenging in multiple perspectives. Particularly, many existing LLMs are initially introduced with a limited size of context (*e.g.*, 2K for Llama-1 Touvron et al. [2023a], 4K for Llama-2 Touvron et al. [2023b], 8K for Llama-3 [2]). Although the initial short-LLM can be fine-tuned to establish a much longer context, it is likely to take substantial costs; and more seriously, it is extremely resource-consuming to deploy the long-LLMs [Kaplan et al., 2020]. The continually training may also compromise the LLMs' general capability over short contexts [Liu et al., 2023, Li et al., 2023a]. In fact, it remains an open problem to explore new solutions which may tackle long-context tasks both effectively and efficiently.

In this paper, we argue that most long-context tasks are *short-context solvable*. That is to say, the long-context tasks, despite associated with long-sequence inputs, can be addressed by merely working with short-contexts in a strategic way. For example, the reading comprehension or summarization of a book can be solved based on the extraction of necessary key facts from the book. The above argument is akin to the working patterns of human beings and modern computers, where arbitrary long-form problems can always be decomposed and solved on top of a limited memory capacity [Adolphs, 1999, Bryant and O'Hallaron, 2011]. However, even if the above argument holds, it is still non-trivial

*Corresponding author.
[2] https://llama.meta.com/llama3/

Preprint. Under review.



(A) Brute-force Solution (B) Naïve RAG (C) LC-Boost: RAG (D) LC-Boost: Divide-and-Conquer

Figure 1: **Illustration for LC-Boost**. The LLM is prompted to reason for how to access to proper context and how to utilize the accessed context to solve the task. **Toy Examples.** (A) Brute-force solution. Despite correctness, it is unnecessarily expensive due to the processing of the entire context simultaneously. (B) Naive RAG. It is hard to handle problems like information aggregation, which leads to the incomplete answer. (C) LC-Boost leverages RAG to tackle the problem, which produces the correct answer in a small cost. (D) LC-Boost processes the long-context via sequential scan, which correctly solves the problem based on the comprehensively collected information.

to solve the long-context tasks purely based on short contexts. This is because different tasks call for distinct ways of accessing and utilizing information from the long context; therefore, there can hardly be any fixed rules to handle all possible situations. To address this challenge, we propose a method, called **LC-Boost**, where short-LLMs are employed to solve general long-context tasks in a bootstrapping manner. LC-Boost operates with two critical reasoning steps. One is the reasoning of **Access**, where the LLM prompts itself to plan for how to access the appropriate part of context within the input. The other one is the reasoning of **Utilize**, where the LLM figures out how to make effective use of the accessed context. Thanks to the above design, LC-Boost is able to adaptively handle diversified long-context tasks according to their unique nature. For example, given a knowledge-grounded QA problem, the LLM may directly access to the knowledgable context through retrieval, and generate the answer in the form of RAG. Besides, it may sequentially scan the long context chunk-by-chunk if the task calls for the aggregation of specific information from the entire input.

The following toy examples are presented to better illustrate the mechanism of LC-Boost (Figure 1). Particular, there are two common approaches to tackle long-context problems: (A) the brute-force method based on long-LLMs, (B) the surrogate methods, like RAG Xu et al. [2023a]. Despite being straightforward, the brute-force method is likely to incur huge unnecessary costs as the problem could be directly solved by simple surrogate methods, like RAG. On the other hand, although the surrogate methods may help in certain cases, they are likely to become useless in other situations. For instance, the RAG-based methods are inappropriate to handle information aggregation problems, as showcased in (B). In contrast, LC-Boost is able to handle general long-context tasks thanks to the proper reasoning of how to access and utilize the long-context information based on each specific task. As shown in (C), it can directly access to the needed information via retrieval and generate the answer based on RAG. Meanwhile, it can also process the entire context in a divide-and-conquer manner, which will fully collect the information and solve the problem presented in (D).

We perform comprehensive experiments for LC-Boost, including both popular real-world long-context problems, like question-answering and summarization of long documents, and a wide variety of synthetic tasks. In our experiments, LC-Boost is able to achieve equivalent performances as the brute-force methods based on strong long-LLMs, *e.g.*, GPT-4-128K. In many cases, its performances can even notably surpass the brute-force methods, probably due to the elimination of distracting

context. Besides, our experiments also underscore the importance of reasoning and adaptability, as LC-Boost outperforms all short-LLM surrogates with predefined access and utilization of context.

To summarize, our paper makes the following contributions. (1) We identify the research problem of solving long-context problems with short-LLMs. To the best of our knowledge, it is the first study of its kind, which is important to not only address the problem itself but also meaningful to the sustainability and energy-efficient running of AI industry in a broader sense. (2) We propose a novel framework LC-Boost, which is able to adaptively handle general long-context tasks based on the reasoning of how to access and utilize the long context. (3) We empirically verify the effectiveness of LC-Boost based on its superior performances achieved from low resource-consumption.

2 LC-Boost

2.1 Preliminaries

LLMs can be succinctly defined as $\mathcal{Y} = \gamma(q)$, where $\gamma(\cdot)$ represents a selected LLM, q denotes a user query, and $\mathcal{Y}$ refers to the answer produced by the LLMs. As highlighted in many previous studies, *e.g.*, [Ji et al., 2023, Lewis et al., 2020, Shuster et al., 2021], the knowledge embedded in an LLM's parameters is static and, consequently, often fails to adequately address user queries requiring up-to-date or in-depth knowledge. To address this limitation, we can introduce external knowledge (refer to as context $\mathcal{X}$) into the LLMs. Additionally, tasks involving information aggregation (*e.g.*, summarization) also take a context $\mathcal{X}$ as input along with task instructions q. Thus, we can generally define the model's generation process w.r.t. a context $\mathcal{X}$ as: $\mathcal{Y} = \gamma(q, \mathcal{X})$.

As discussed in Section 1, in many scenarios, the context $\mathcal{X}$ is a long sequence, necessitating that LLMs manage long contexts. However, most existing LLMs were originally introduced with limited context sizes (*e.g.*, 4K). Consequently, these models are unable to process inputs that exceed their capacity without truncation. In this paper, we characterize such scenarios as *long-context problem*. It involves LLMs processing inputs that notably surpass their inherent context limitations, which can be formally described by:

$$\mathcal{Y} = \gamma(q, \mathcal{X}) \quad \text{s.t.} |\mathcal{X}| \gg L, \tag{1}$$

where L denotes the native context length limit of the LLM. The most straightforward way to address the long-context problem is to increase the LLMs' context length L, mitigating the challenges of long contexts. In this paper, we instead explore solving long-context tasks using short-context LLMs (*e.g.*, 4K) without increasing the model's context length L.

2.2 Pilot Study: Are Most Long-Context Tasks Short-Context Solvable?

Despite the potential for fine-tuning LLMs to handle much longer contexts, this approach incurs substantial costs. Additionally, directly processing long contexts during the inference stage exponentially increases computing resource consumption, which is not environmentally friendly. In the following, we conduct a pilot study from both theoretical and empirical perspectives to explore the question: Are most long-context tasks solvable with short contexts?

Theoretical Analysis Suppose we have an input variable $\mathcal{X}$ and an output variable $\mathcal{Y}$, the relevant part of $\mathcal{X}$ given $\mathcal{Y}$ is denoted by $\tilde{\mathcal{X}}$. An ideal $\tilde{\mathcal{X}}$ should capture all relevant features of the original input variable $\mathcal{X}$ in relation to $\mathcal{Y}$. In other words, the optimal $\tilde{\mathcal{X}}$ represents the simplest mapping of $\mathcal{X}$ that accurately preserves the mutual information $I(\mathcal{X}; \mathcal{Y})$. We therefore propose a Markov chain $\mathcal{X} \rightarrow \tilde{\mathcal{X}} \rightarrow \mathcal{Y}$. According to the data processing inequality (DPI), we have $I(\mathcal{X}; \tilde{\mathcal{X}}) \geq I(\mathcal{X}; \mathcal{Y})$, with equality holding if and only if $\tilde{\mathcal{X}}$ constitutes a *sufficient statistics* [Cover, 1999, Tishby and Zaslavsky, 2015]. This suggests that, in an optimal setting, we can always find a subset $\tilde{\mathcal{X}} \subseteq \mathcal{X}$ that provides information at least as useful for generating the output $\mathcal{Y}$ as the full context $\mathcal{X}$.

In practical scenarios, obtaining the optimal $\tilde{\mathcal{X}}$ is challenging due to various factors, such as empirical errors Mohri et al. [2018]. Thus, we can only estimate $\tilde{\mathcal{X}}$. Estimating $\tilde{\mathcal{X}}$ directly from $\mathcal{X}$ might be challenging if $\mathcal{X}$ defines a large variable space. In this situation, we propose decomposing the original input variable $\mathcal{X}$ into a series of subsets, $\mathcal{X} = \{\mathcal{X}_1, \cdots, \mathcal{X}_n\}$ and process each subset variable

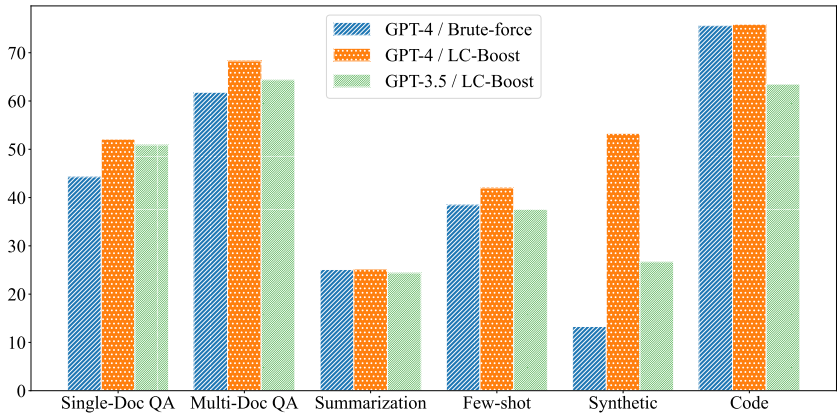


Figure 2: Pilot Study Across Various Tasks: In the Brute-force setting, the entire context is processed by GPT-4-128K. In the LC-Boost setting, the maximum context length is restricted to 4K, and LC-Boost is utilized to solve the long-context problem with short context.

separately. Thus, according to the *chain rule for mutual information* Cover [1999], we have:

$$I(\mathcal{X}, \tilde{\mathcal{X}}) = I(\mathcal{X}_1, \cdots, \mathcal{X}_n; \tilde{\mathcal{X}}) = I(\mathcal{X}_1; \tilde{\mathcal{X}}) + \sum_{i=2}^{n} I(\mathcal{X}_i; \tilde{\mathcal{X}} | \mathcal{X}_1, \cdots, \mathcal{X}_{i-1}), \tag{2}$$

which indicates that the mutual information $I(\mathcal{X}, \tilde{\mathcal{X}})$ can be understood as the sum of the mutual information of each subset $\mathcal{X}_i$ and $\tilde{\mathcal{X}}_i$ given all previous subsets.

In the scenario of Eq. 1, the variable $\mathcal{X}$ represents a long context and the variable $\mathcal{Y}$ is the output answer produced by a LLM. Thus, $\tilde{\mathcal{X}}$ can be interpreted as the *minimal necessary context* from the long context $\mathcal{X}$ given the output answer $\mathcal{Y}$. Inspired by Eq. 2, we can estimate an optimal $\tilde{\mathcal{X}}$ using decomposed shorter contexts $\{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$. Thus, $I(\mathcal{X}; \tilde{\mathcal{X}})$ can be computed by processing each subset $\mathcal{X}_i$ individually. However, as the number of subsets n increases, accounting for all preceding subsets becomes computationally demanding. To alleviate this burden, we propose reducing the number of conditional subsets considered by replacing the entire sequence of previous subsets with a compressed surrogate $\hat{\mathcal{X}}_i$, which is iteratively derived using a compression function $\hat{\mathcal{X}}_i = g(\hat{\mathcal{X}}_{i-1}, \mathcal{X}_{i-1})$. Consequently, Eq. 2 can be reformulated as follows:

$$I(\mathcal{X}, \tilde{\mathcal{X}}) = I(\mathcal{X}_1, \cdots, \mathcal{X}_n; \tilde{\mathcal{X}}) \simeq I(\mathcal{X}_1; \tilde{\mathcal{X}}) + \sum_{i=2}^{n} I(\mathcal{X}_i; \tilde{\mathcal{X}} | \hat{\mathcal{X}}_i)). \tag{3}$$

The equality can be upheld under two specific conditions: (1) the decomposed variables $\{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$ are mutually independent, and (2) the compression function $g(\cdot)$ is optimally designed, ensuring that the compressed surrogate $\hat{\mathcal{X}}_i$ encapsulates all relevant information from the preceding subsets with respect to $\tilde{\mathcal{X}}$. Otherwise, $I(\mathcal{X}, \tilde{\mathcal{X}})$ can only be approximately estimated.

Empirical Analysis To empirically assess the accuracy of estimating the minimal necessary context $\tilde{\mathcal{X}}$ using decomposed short contexts $\{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$, we conduct pilot experiments across various tasks requiring long contexts. Specifically, we utilize GPT-4-128K to perform these tasks in two settings: (1) feeding the entire long context into GPT-4-128K in a brute-force manner, instructing the model to directly produce the output answer, and (2) decomposing the full context into short contexts and applying the methods defined in Eq. 3 to approximate $\tilde{\mathcal{X}}$, which then guides the model to produce the final output (the LC-Boost setting).

Figure 2 presents the experiment results, which generally indicate that LC-Boost consistently performs as well as or better than the brute-force setting. In particular, for tasks such as QA, few-shot learning, and synthetic tasks, LC-Boost outperforms the brute-force setting. This is because the decomposed short contexts for these tasks are more likely to be mutually independent given the input query which can be adequately supported by a few extracted contexts from the long context. By precisely locating these supported context, it can filter out irrelevant context of $\mathcal{X}$ that might otherwise undermine task

performance. For tasks like summarization and code completion, the inherent properties of these tasks require considering the mutual dependencies among all decomposed short contexts, making the LC-Boost setting more challenging. However, as discussed in Eq. 3, when the compression function $g(\cdot)$ is optimal, we can achieve the optimal $\tilde{\mathcal{X}}$. GPT-4 serves as such a strong compression function, ensuring that the compressed surrogate $\hat{\mathcal{X}}_i$ is well-estimated. Consequently, in these tasks, LC-Boost achieves performance that is equal to or slightly better than the brute-force setting.

Through theoretical analysis, we can posit that long-context tasks are short-context solvable if we can estimate a better minimal necessary context $\tilde{\mathcal{X}}$ from the decomposed short contexts $\{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$ than from the long context $\mathcal{X}$. Empirical analysis supports this assumption, demonstrating that in most cases, the estimation error of deriving $\tilde{\mathcal{X}}$ from the long context $\mathcal{X}$ is often larger than from the decomposed short contexts $\{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$. This indicates that using short contexts can be comparatively more advantageous than using the full context. Therefore, we can validate our argument in Section 1: **most long-context tasks, if not all, are short-context solvable.**

2.3 The Proposed Method: LC-Boost

We propose a method called LC-Boost, which utilizes short LLMs to solve general long-context tasks. LC-Boost begins with an input query q and a long context $\mathcal{X}$, with the goal of producing an output answer $\mathcal{Y}$. Since the underlying LLM in LC-Boost has a limited context size (we limit LC-Boost working with 4K context length), directly generating the output answer $\mathcal{Y}$ is infeasible for long-context tasks. To address this, we propose solving long-context tasks by strategically understanding the decomposed short contexts $\mathcal{X} = \{\mathcal{X}_1, \cdots, \mathcal{X}_n\}$. From these short contexts, we aim to extract the minimal necessary context $\tilde{\mathcal{X}}$ to support the generation of the output answer $\mathcal{Y}$.

this is the difference, they do it to answer 1 query from the long context, we are doing it to answer multiple queries spanning throughout the long context. in our problem identification, we can also indicate which queries were answered wrongly (verifying "lost in the middle" problem)

LC-Boost achieves this goal through a decision-making process involving iterative interactions between LC-Boost and the decomposed short contexts $\{\mathcal{X}_1, \cdots, \mathcal{X}_n\}$ with respect to the input query q. In the process, LC-Boost interact with each short context $\mathcal{X}_i$, employing two types of actions: information access and information utilization.

We denote an action at time step i by a_i and denote the relevant context LC-Boost obtains from the i-th short context $\mathcal{X}_i$ by $\tilde{\mathcal{X}}_i$ The action a_i is predicted by considering the current short context $\mathcal{X}_i$, the input query q, as well as all previous extracted relevant information $\tilde{\mathcal{X}}_{1:i-1}$: $a_i = \gamma(q, \mathcal{X}_i | \tilde{\mathcal{X}}_{1:i-1})$, where $\gamma(\cdot)$ denotes LC-Boost's underlying LLM.

Predicting the action a_i in a continuous space is challenging as it requires the underling model to reason about highly implicit relations among the input query, the current context, and the previous contexts. Therefore, we define a discrete action space $\mathcal{A}$ comprising: (1) `[Task Understanding]`: analyzing the query and task for initialization; (2) `[Retrieve]`: accessing text evidence by a retrieval method; (3) `[Move]`: accessing the next short text context directly; These two are information access actions which define the LC-Boost's trajectory to access short contexts. (4) `[Append]`: generating relevant context $\tilde{\mathcal{X}}_i$ independently, denoting by $\tilde{\mathcal{X}}_i = a_i(\mathcal{X}_i)$; (5) `[Merge]`: generating relevant context $\tilde{\mathcal{X}}_i$ with respect to previous extracted relevant information, denoting by $\tilde{\mathcal{X}}_i = a_i(\mathcal{X}_i | \tilde{\mathcal{X}}_{1:i-1})$; (6) `[Answer]`: answering the user query and returning; (7) `[Aggregation]`: aggregating all relevant information and returning. We define our LC-Boost frame in Algorithm 1.

Though the pre-defined action space $\mathcal{A}$ comprises only seven actions, LC-Boost serves as a general framework sufficient for solving most long-context tasks. This effectiveness is based on the following reasons: **(1) Flexible accessibility:** By utilizing both `[Retrieve]` and `[Move]` actions, LC-Boost can access any short context $\mathcal{X}_i \in \mathcal{X}$ in a flexible trajectory, avoiding the need to browse the entire long context. This makes the information accessing process more efficient. **(2) Accurate information acquisition:** Through the `[Append]` and `[Merge]` actions, LC-Boost can either independently extract relevant information from the current short context, appending it to previously extracted information, or merge the current relevant information into the previous relevant information. This capability allows LC-Boost to acquire relevant information in a compatible manner, making it adaptable to many knowledge-intensive tasks. and **(3) Dynamic answering:** Using the `[Answer]` and `[Aggregate]` actions, LC-Boost can dynamically utilize the acquired relevant information to produce the target form of the answer (e.g., a short answer for QA tasks via the `[Answer]` action, or a long answer for summarization tasks via the `[Aggregate]` action).

Algorithm 1 LC-Boost Framework

Input: Input query q, long context $\mathcal{X}$
Output: Answer $\mathcal{Y}$
Decompose long context $\mathcal{X} \leftarrow \{\mathcal{X}_1, \cdots, \mathcal{X}_n\}$
Initialize extracted relevant context $\tilde{\mathcal{X}}_0 \leftarrow$ None
Perform `[Task Understanding]`
while $i \leq n$ **do**
  Select an action $a_i \leftarrow a_i = \gamma(q, \mathcal{X}_i | \tilde{\mathcal{X}}_{1:i-1}),\ a_i \in \mathcal{A}$
  if a_i is `[Move]` **then** $i \leftarrow i + 1$, continue
  if a_i is `[Retrieve]` **then** retrieve evidence from $\mathcal{X} = \{\mathcal{X}_1, \cdots, \mathcal{X}_n\}$
  if a_i is `[Append]` **then** generate relevant context by $\tilde{\mathcal{X}}_i = a_i(\mathcal{X}_i)$
  if a_i is `[Merge]` **then** generate relevant context by $\tilde{\mathcal{X}}_i = a_i(\mathcal{X}_i | \tilde{\mathcal{X}}_{1:i-1})$
  if $a_i \in \{$`[Answer]`, `[Aggregation]`$\}$ **then** generate answer $\mathcal{Y} = \gamma(q, \tilde{\mathcal{X}}_{1:i})$, **break**
  $i \leftarrow i + 1$
end while
return answer $\mathcal{Y}$

In our pilot study depicted in Figure 2, we observe that while GPT-3.5 serves as an inferior foundation model compared to GPT-4, it still demonstrates significant effectiveness when incorporated with LC-Boost. Given considerations of efficiency and cost-effectiveness, we employ GPT-3.5 as the foundation model for LC-Boost in the subsequent experiments. Besides, we show the prompts used in LC-Boost in Appendix B.

3 Experiments

3.1 Experiment Settings

We evaluate LC-Boost and baseline models on 12 datasets, including: (1) Single-Doc QA: NarrativeQA [Kočiský et al., 2017], Qasper [Dasigi et al., 2021], and MultiFieldQA [Bai et al., 2023]. (2) Multi-Doc QA: HotpotQA [Yang et al., 2018], 2WikiMQA [Ho et al., 2020], and MuSiQue [Trivedi et al., 2022]. (3) Summarization: GovReport [Huang et al., 2021] and MultiNews [Fabbri et al., 2019]. (4) Few-shot Learning: SAMSum [Gliwa et al., 2019]. (5) Synthetic Task: Passage Count [Bai et al., 2023] and Self-Constructed Dataset. (6) Code Completion: LCC [Guo et al., 2023]. More details about the evaluation datasets and metrics are introduced in Appendix A.

doesnt use squad

We compare our LC-Boost with three types of models: (1) Short LLMs (defined as with context length $<$ 32K): Llama2-7B-Chat-4K [Touvron et al., 2023b], Llama3-8B-Instruct-8K and Vicuna-v1.5-7B-16K [Chiang et al., 2023]; (2) Long LLMs (defined as with context length $\geq$ 32K): LongChat-v1.5-7B-32K [Li et al., 2023b], Mistral-7B-Instruct-v0.2-32K [Jiang et al., 2023a], Llama3-8B-80K Zhang et al. [2024b], Phi-3-mini-128K [Abdin et al., 2024] and Yi-9B-200K [AI et al., 2024]; (3) Closed-Source LLMs: DeepSeek-v2 (236B MoE model, ranks top-tier in MT-Bench) [DeepSeek-AI, 2024], Claude-3-Haiku[3] and GPT-3.5-turbo-16K[4]. In the experiments, if the context length exceed the model's length limit, following Bai et al. [2023], we truncate the context from the middle since the front and end of the context may contain crucial information. We provide further implementation details in Appendix B.

3.2 Main Results

Table 1 shows the overall experimental results for all models across all tasks. From the table, we derive several key findings: **First**, LC-Boost, with a context length of 4K, outperforms all baseline models in all tasks except for the Code Completion task. This result verifies LC-Boost's capability to effectively solve long-context tasks by strategically processing decomposed short contexts. **Second**, long LLMs generally perform better than short LLMs, indicating the effectiveness of fine-tuning LLMs to adapt to long contexts. However, the performance of long LLMs is not consistently stable across different tasks. For example, Yi-9B-200K excels in the Code Completion task but does not

[3] `https://www.anthropic.com/claude`
[4] `https://platform.openai.com/docs/models`

Table 1: Main experiment results. The best results are in bold and the secondary results are marked with underline. We report the average scores (%) on the main tasks. The detailed scores over all dataset are shown in Table 3.

Models	Single-Doc	Multi-Doc	Summ.	Few-shot	Synthetic	Code
Short LLMs (Context Length $<$ 32K)						
Llama2-7B-Chat-4K	24.9	22.5	26.6	40.7	6.3	52.4
Llama3-8B-Instruct-8K	37.3	36.0	26.5	42.7	15.0	57.5
Vicuna-v1.5-7B-16K	28.0	18.6	27.5	40.8	8.9	51.0
Long LLMs (Context Length $\geq$ 32K)						
LongChat-v1.5-7B-32K	28.7	20.6	28.6	34.2	6.8	53.0
Mistral-7B-Instruct-v0.2-32K	31.9	26.0	29.3	<u>43.0</u>	14.0	55.4
Llama3-8B-80K	43.6	43.1	30.2	42.9	19.6	53.6
Phi-3-mini-128K	33.5	38.2	28.8	36.0	19.9	<u>60.1</u>
Yi-9B-200K	29.6	38.7	28.4	14.6	6.5	**72.1**
Closed-Source LLMs						
DeepSeek-v2 (32K)	37.6	<u>49.1</u>	<u>30.8</u>	39.3	14.5	37.0
Claude-3-Haiku (200K)	<u>41.9</u>	45.4	30.1	7.2	<u>25.5</u>	16.9
GPT-3.5-turbo-16K	39.8	38.7	28.1	41.7	18.7	54.7
LC-Boost (4K)	**47.8**	**56.4**	**31.8**	**44.1**	**27.5**	59.0

show consistent performance in other tasks such as single-doc QA, few-shot learning, and synthetic tasks. This inconsistency suggests that adapting LLMs to long contexts may compromise their general abilities. **Last**, LC-Boost consistently surpasses its underlying LLM, GPT-3.5-turbo-16K, across all tasks by a notable margin. This demonstrates that LC-Boost can achieve improved performance while simultaneously reducing resource costs, making LC-Boost an environmentally friendly method.

3.3 Ablation Study: Dynamic is Important

we will also include the ablation study of our method

To investigate the necessity of LC-Boost's design, we conduct ablation studies by changing LC-Boost's action space $\mathcal{A}$, resulting in different information acquisition strategies. We experiment with the following settings: (1) `[Retrieve]` only: Directly retrieve the most relevant short context. (2) `[Merge]` only: Sequentially process all short contexts while considering the previously processed context. (3) `[Append]` only: Sequentially process all short contexts independently. (4) `[Merge]` & `[Move]`: Selectively process short contexts while considering the already processed context. (6) `[Append]` & `[Move]`: Selectively process short contexts independently. (7): `[Retrieve]` & `[Move]`: Retrieve the top-k relevant short contexts and selectively process a few of them. (8): Brute-force: Directly produce the answer based on the entire long context. (9) Random: For each short context, randomly select an action. Based on the acquired information from each strategy, LC-Boost then selects either the `[Answer]` or `[Aggregation]` action to produce the final answer.

Figure 3 illustrates the results, from which we find that: (1) Compared to fixed processing strategies, LC-Boost customizes the action trajectory for each query, resulting in notable performance improvements. This finding emphasizes the importance of the dynamic capabilities of LC-Boost. (2) LC-Boost is particularly effective in single-doc QA and multi-doc QA tasks, as it can accurately select the minimal necessary context required to answer the input query, filtering out irrelevant information from the long context. (3) In the few-shot learning task, LC-Boost does not significantly outperform the fixed strategies. This is attributed to the numerous in-context examples provided within the task, which offer substantial guidance, thus diminishing the impact of the number of in-context examples on the final performance.

3.4 Case Study: Model Behavior Analysis on Self-Construct Dataset

In Table 2, we present two case studies from the self-constructed dataset. These cases are particularly challenging as they require reasoning across the entire long context. Despite having sufficient context size, LLMs struggle to generate correct responses. In contrast, LC-Boost dynamically customizes solutions for each case, thereby effectively solving the problems using a shorter context length.

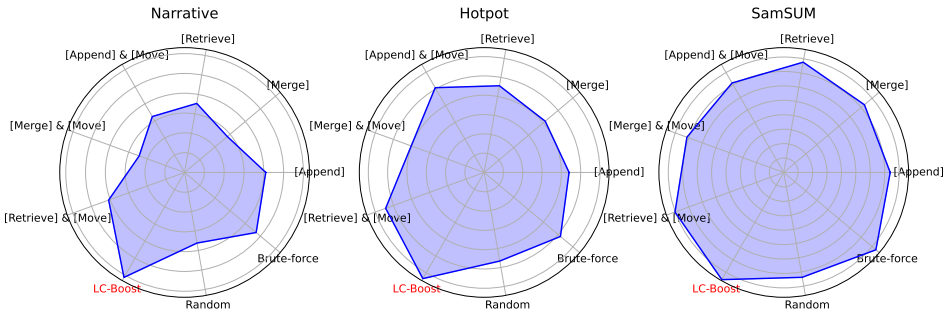


Figure 3: Performance comparison on different context processing strategies in the ablation study. NarrativeQA (left) is a single-doc QA task. HotpotQA (middle) is a multi-doc QA task. SamSUM (right) is a few-shot learning task.

Table 2: Case study on the self-constructed dataset. Correct answers are marked in teal, incorrect answers in red, and ambiguous answers in orange.

Query: How many papers in ACL 2023 only have one author? **Context**: Full accepted paper list in ACL 2023 main conference. (Context length: 45K) **Ground-truth target**: 8 papers
Phi-3-mini-128K: 11 papers **GPT-3.5-turbo-16K**: 0 papers **Claude-3-Haiku-200K**: 1 papers (Acc. Score: 0)
LC-Boost's action trajectory: `[Task Reasoning]` → `[Append]` → ··· → `[Append]` → `[Aggregation]` **LC-Boost**: 8 papers (Acc. Score: 1)
Query: List all people names that are petrified, separated by comma. **Context**: Full content of Harry Potter and the Chamber of Secrets. (Context length: 122.6K) **Ground-truth target**: Colin Creevey, Justin Finch-Fletchley, Penelope Clearwater, Hermione Granger
Phi-3-mini-128K: Hermione Granger, Ginny Weasley, Mrs Norris (F1-Score: 0.29) **GPT-3.5-turbo-16K**: Colin Creevey, Mrs Norris (F1-Score: 0.33) **Claude-3-Haiku-200K**: Nick, Hermione, Ron (F1-Score: 0.18)
LC-Boost's action trajectory: `[Task Reasoning]` → `[Move]` → ··· → `[Merge]` → `[Aggregation]` **LC-Boost**: Colin Creevey, Penelope Clearwater, Hermione Granger, Nick, Mrs Norris (F1-Score: 0.71)

For the first query, LC-Boost performs `[Append]` or `[Move]` actions across all short context along with a rewritten query, "Extract paper information in the following list that have only one author," derived via `[Task Reasoning]`. After processing all short contexts, LC-Boost employs the `[Aggregation]` action to compile the final answer. This approach simplifies the task compared to directly extracting a numeric answer from the entire long context, mimicking the human process of reading comprehension and thereby producing accurate results.

In the second case, the query necessitates conditional reasoning on each short context. As highlighted in previous research [Liu et al., 2023], reasoning directly from the entire context risks losing crucial information, particularly in the middle of the long context. Thus LLMs tend to miss key details such as people's names. LC-Boost addresses this issue by processing only one short context at a step where it extracts information from arbitrary position of the long text with equal accuracy. Additionally, answers marked in orange include non-human names (*e.g.*, cat, ghost) that are misconstrued as people, illustrating a common challenge where models fail to differentiate in-depth entity properties.

3.5 Context be Short, Energy be Saved!

Recently, we have witnessed the remarkable success of LLMs, which are becoming an indispensable part of our daily lives. We believe that in the near future, LLMs will become as ubiquitous as electricity or gas supply, serving as fundamental infrastructure in human society. At that point, the energy consumption of LLMs will emerge as a significant environmental concern. Therefore, it is imperative for the research community to focus on reducing the energy consumption associated with these models. Figure 4 presents an analysis of energy consumption, comparing the brute-force

approach with our LC-Boost method. The y-axis is measured in Joules. The theoretical energy consumption is estimated for 7B LLMs across varying context lengths. We roughly estimate the energy consumption using the formula $\left(\frac{\text{Total Float Operation}}{312 \text{ TFLOPS}}\right) \times 400W$, assuming the use of an A100 GPU with a compute capability of 312 TFLOPS for BFLOAT16 operations and a maximum TDP of 400W[5]. The practical energy consumption is estimated by recording the GPU time and GPU power during inference with different context lengths. We use a Llama2-7B-128K [Peng et al., 2023] and a Llama2-7B-chat-4K [Touvron et al., 2023a] for the brute-force setting and LC-Boost, respectively. Figure 4 clearly indicates that longer context lengths significantly increase energy consumption with the brute-force method, especially evident in practical measurements. This difference is primarily due to the need to distribute sequence activation tensors across multiple GPUs in practical experiment, with tensor I/O exacerbating inference latency and thereby inflating energy costs. In contrast, our LC-Boost method, working with 4K context lengths, shows only a mild increase in energy consumption across contexts, thereby confirming its energy efficiency while maintaining comparable or superior performance on long-context tasks. We also provide an analysis on token consumption in Appendix C.

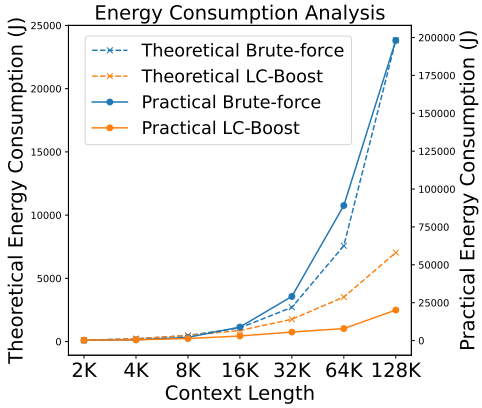


Figure 4: Energy consumption analysis.

4 Related Works

Dealing with long contexts is a fundamental research problem for LLMs, as many real-world applications involve long-context inputs [Li et al., 2023a, Fu et al., 2024]. The most direct approach to address long-context tasks is to increase the working context size of LLMs [Abdin et al., 2024, AI et al., 2024, Li et al., 2023a, Cai et al., 2024]. A year ago, significant research efforts focused on extending the working context size of LLMs from 4K to 32K [Jiang et al., 2023a, Li et al., 2023b, Chen et al., 2023a, Du et al., 2022]. Currently, many popular open-source and close-source LLMs still operate with a context size under 32K [Touvron et al., 2023a, OpenAI, 2023], such as GPT-3.5-turbo, which has a 16K context length. Recently, research has shifted towards extending LLMs' working context to the million-level. Notably, GPT-4 was updated to a 128K context length not long ago, and the newly released GPT-4o also operates with a 128K context. Moreover, several recent open-source LLMs have been introduced with context lengths exceeding 100K, for example, the Yi series model supports up to 200K [AI et al., 2024], and the Phi-3 model operates with 128K [Abdin et al., 2024].

add all of these in our related works as well

Instead of merely increasing the context length, another approach to address long-context tasks involves extracting a short surrogate context from the full context. This includes techniques like retrieval-augmented generation (RAG) and context refinement methods [Izacard and Grave, 2021a, Gao et al., 2024, Wang et al., 2023, Qian et al., 2024]. However, many of these methods utilize task-specific strategies to manage the long context. For instance, RAG methods often deploy retrievers to select relevant context chunks as supporting evidence [Izacard and Grave, 2021b, Xu et al., 2023b, Jiang et al., 2023b]. Recent studies have criticized the chunking process in RAG for undermining the semantic coherence of the long context and have proposed chunking-free methods to refine the long context into a concise surrogate context [Qian et al., 2024, Luo et al., 2024]. Furthermore, some studies have also explored sequential processing strategies, such as Ratner et al. [2022] and Xu et al. [2023a], to sequentially process the context in a manner that preserves its integrity.

Lastly, reasoning-based methods also show significant potential for addressing long context tasks [Nakano et al., 2022, Yang et al., 2023, Driess et al., 2023]. These methods predominantly employ a decision-making process to navigate through the long context sequentially, utilizing reasoning techniques such as in-context learning [Dong et al., 2022], chain-of-thought [Wei et al., 2022], and self-reflection [Shinn et al., 2023]. In this paper, LC-Boost incorporates a decision-making process that dynamically customizes the action trajectory for each query, thereby offering considerable flexibility in accessing and leveraging information to produce the final output answer.

[5]The calculation of total float operations is based on the method outlined in `https://www.harmdevries.com/post/context-length/`

5 Conclusion

In this paper, we argue that most long-context tasks are short-context solvable, and we validate this claim through both theoretical and empirical analysis. We propose a method called LC-Boost to solve long-context tasks by decomposing the long context into short contexts and processing them using a decision-making process. We conduct experiments on 12 datasets to compare LC-Boost with long LLMs and other baseline models. Empirical results verify LC-Boost's effectiveness in solving long-context tasks. Additionally, we discuss the energy consumption of LC-Boost versus long LLMs, demonstrating that LC-Boost can achieve comparable performance with significantly less energy consumption. In Appendix D, we also discuss the limitations and broader impact of this paper.

References

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*, 2023.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020.

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, 2023.

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

Ralph Adolphs. Social cognition and the human brain. *Trends in cognitive sciences*, 3(12):469–479, 1999.

Randal E Bryant and David Richard O'Hallaron. *Computer systems: a programmer's perspective*. Prentice Hall, 2011.

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*, 2023a. doi: 10.48550/arxiv.2310.03025. Experimental.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-Augmented Generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474, 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf`.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.320. URL `https://aclanthology.org/2021.findings-emnlp.320`.

Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle, 2015.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The narrativeqa reading comprehension challenge, 2017.

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 *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, 2021.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering, 2018.

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. doi: 10.18653/v1/2020.coling-main.580. URL https://aclanthology.org/2020.coling-main.580.

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.

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. doi: 10.18653/v1/2021.naacl-main.112. URL https://aclanthology.org/2021.naacl-main.112.

Alexander R. Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R. Radev. Multi-news: a large-scale multi-document summarization dataset and abstractive hierarchical model, 2019.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In Lu Wang, Jackie Chi Kit Cheung, Giuseppe Carenini, and Fei Liu, editors, *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5409. URL https://aclanthology.org/D19-5409.

Daya Guo, Canwen Xu, Nan Duan, Jian Yin, and Julian McAuley. Longcoder: A long-range pre-trained language model for code completion, 2023.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Dacheng Li, Rulin Shao, Anze Xie, Ying Sheng, Lianmin Zheng, Joseph E. Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. How long can open-source llms truly promise on context length?, June 2023b. URL https://lmsys.org/blog/2023-06-29-longchat.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023a.

Peitian Zhang, Ninglu Shao, Zheng Liu, Shitao Xiao, Hongjin Qian, Qiwei Ye, and Zhicheng Dou. Extending llama-3's context ten-fold overnight, 2024b.

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Amit Garg, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee,

Yuanzhi Li, Chen Liang, Weishung Liu, Eric Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Xia Song, Masahiro Tanaka, Xin Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Michael Wyatt, Can Xu, Jiahang Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 technical report: A highly capable language model locally on your phone, 2024.

01. AI, :, Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Tao Yu, Wen Xie, Wenhao Huang, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Pengcheng Nie, Yuchi Xu, Yudong Liu, Yue Wang, Yuxuan Cai, Zhenyu Gu, Zhiyuan Liu, and Zonghong Dai. Yi: Open foundation models by 01.ai, 2024.

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

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. In *The Twelfth International Conference on Learning Representations*, 2023.

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

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, et al. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*, 2024.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models. In *The Twelfth International Conference on Learning Representations*, 2023a.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335, 2022.

OpenAI. Gpt-4 technical report. `https://cdn.openai.com/papers/gpt-4.pdf`, 2023.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering, 2021a.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey, 2024.

Zhiruo Wang, Jun Araki, Zhengbao Jiang, Md Rizwan Parvez, and Graham Neubig. Learning to filter context for retrieval-augmented generation, 2023.

Hongjin Qian, Zheng Liu, Kelong Mao, Yujia Zhou, and Zhicheng Dou. Grounding language model with chunking-free in-context retrieval, 2024.

Gautier Izacard and Edouard Grave. Distilling knowledge from reader to retriever for question answering. In *International Conference on Learning Representations*, 2021b. URL `https://openreview.net/forum?id=NTEz-6wysdb`.

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. In *The Twelfth International Conference on Learning Representations*, 2023b.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*, 2023b. URL `https://arxiv.org/pdf/2305.06983`.

Kun Luo, Zheng Liu, Shitao Xiao, and Kang Liu. Bge landmark embedding: A chunking-free embedding method for retrieval augmented long-context large language models, 2024.

Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. Parallel Context Windows Improve In-Context Learning of Large Language Models. *arXiv*, 2022. doi: 10.48550/arxiv.2212.10947. Window.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. Webgpt: Browser-assisted question-answering with human feedback, 2022.

Hui Yang, Sifu Yue, and Yunzhong He. Auto-gpt for online decision making: Benchmarks and additional opinions. *arXiv preprint arXiv:2306.02224*, 2023.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=_VjQlMeSB_J`.

Noah Shinn, Beck Labash, and Ashwin Gopinath. Reflexion: an autonomous agent with dynamic memory and self-reflection. *arXiv preprint arXiv:2303.11366*, 2023.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2023b.

Table 3: Main experiment results. The best results are in bold and the secondary results are marked with underline. We report the average scores (%) on all tasks.

Model	Narrative	Qasper	MultiField	Hotpot	MuSiQue	2Wiki
Short LLMs (Context Length $<$ 32K)						
Llama2-7B-Chat-4K	18.7	19.2	36.8	25.4	9.4	32.8
Llama3-8B-Instruct-8K	21.5	43.0	47.5	47.3	23.3	37.5
Vicuna-v1.5-7B-16K	19.4	26.1	38.5	25.3	9.8	20.8
Long LLMs (Context Length $\geq$ 32K)						
LongChat-v1.5-7B-32K	16.9	27.7	41.4	31.5	9.7	20.6
Mistral-7B-Instruct-v0.2-32K	21.6	29.2	47.9	37.7	18.6	21.8
Llama3-8B-80K	28.8	47.4	54.5	55.8	27.4	46.0
Phi-3-mini-128K	21.0	39.4	51.5	48.1	28.2	38.1
Yi-9B-200K	15.6	39.3	33.8	51.4	26.6	38.2
Closed-Source LLMs						
DeepSeek-v2 (32K)	18.3	45.7	48.9	57.7	22.6	**66.9**
Claude-3-Haiku (200K)	30.2	44.0	51.5	51.5	32.5	52.1
GPT-3.5-turbo-16K	23.6	43.3	52.3	51.6	26.9	37.7
LC-Boost (4K)	**30.6**	**50.6**	**62.1**	**63.5**	**42.5**	63.1

Model	GovReport	MultiNews	SAMSum	LCC	PCount	Self
Short LLMs (Context Length $<$ 32K)						
Llama2-7B-Chat-4K	27.3	25.8	40.7	52.4	2.1	10.5
Llama3-8B-Instruct-8K	30.1	27.6	42.7	57.5	8.0	21.9
Vicuna-v1.5-7B-16K	27.9	27.2	40.8	51.0	6.5	11.3
Long LLMs (Context Length $\geq$ 32K)						
LongChat-v1.5-7B-32K	30.8	26.4	34.2	53.0	1.0	12.5
Mistral-7B-Instruct-v0.2-32K	31.7	26.9	43.0	55.4	2.6	25.4
Llama3-8B-80K	32.3	28.1	42.9	53.6	3.5	35.7
Phi-3-mini-128K	32.6	24.9	36.0	60.1	3.2	36.5
Yi-9B-200K	30.3	26.5	14.6	**72.0**	4.2	8.7
Closed-Source LLMs						
DeepSeek-v2 (32K)	**35.2**	26.3	39.3	37.0	**12.7**	16.2
Claude-3-Haiku (200K)	34.1	26.1	7.2	16.9	5.0	46.0
GPT-3.5-turbo-16K	29.5	26.7	41.7	54.7	4.5	32.9
LC-Boost (4K)	34.4	**29.2**	**44.1**	59.0	7.2	**47.7**

A More details of the Datasets

We evaluated all models on 12 datasets, as shown in Table 4. Most of these datasets are provided by the LongBench benchmark [Bai et al., 2023]. Following LongBench, we used F1-score, accuracy, and edit similarity as the evaluation metrics. Additionally, we manually annotated a self-constructed dataset comprising long contexts from practical scenarios, such as the full schedule of the Olympic Games and the complete list of accepted papers at ACL. The queries in the self-constructed dataset involve reasoning over the entire long context. For example, "Who has the most accepted papers at ACL 2023?" These queries require the model to accurately understand the long context and perform reasoning, making them highly challenging. The details of the self-constructed dataset are in Table 5.

B Implementation Details

LC-Boost begins with the `[Task Understanding]` action after receiving the input query and context, using the prompt shown in Table 6. If the task does not include an input query, the two columns "Below is the query" and "{input_query}" are omitted. Besides, for the synthetic task, we use the prompt shown in Table 7 to reformulate the query for better adaptation to LC-Boost. Based on

Table 4: Statistical information of the datasets utilized in this paper.

Dataset	Narrative	Qasper	MultiField	Hotpot	MuSiQue	2Wiki
Num of Samples	200	200	150	200	200	200
Ave. Length	18,409	3,619	4,559	9,151	11,214	4,887
Metric	F1	F1	F1	F1	F1	F1
Dataset	GovReport	MultiNews	SAMSum	PCount	LCC	Self
Num of Samples	200	200	200	200	500	32
Ave. Length	8,734	2,113	6,258	11,141	1,235	39,420
Metric	Rouge-L	Rouge-L	Rouge-L	Accuracy	Edit Sim	F1&Accuracy

Table 5: Data details of the self-constructed dataset.

Source	Length	# Queries	Example Query
Accepted paper list of ACL 2023 Main Conference	44,490	7	Who has the most accepted paper in ACL 2023?
The Diamond Sutra	19,993	3	How many chapters of the Sutra?
Schedule of The 2024 Olympic Games	15,844	9	Which day has the most gold medal events?
Subtitle of The Big Bang Theory S3E14	11,136	6	How long does this episode?
The Little Prince	22,471	4	How many planets does the little prince visit?
Harry Potter and the Chamber of Secrets	122,591	3	How many times has the chamber of secret been opened?

the output of the `[Task Understanding]` action, LC-Boost adopts different strategies to perform the task. Specifically, "option [1]" directs LC-Boost to utilize a retriever to rank all chunks of the long context. In this paper, we employ BGE-Reranker-Large as the retriever Chen et al. [2023b]. For "option [2]" and "option [3]", LC-Boost uses the prompts shown in Table 10 and Table 8 to sequentially process each short context, respectively. After processing each short context, if the output is not "null", the newly summarized context is added to the "previous summarization".

Once all short contexts are processed, LC-Boost aggregates all relevant information to produce the final answer. At this stage, we use the prompt provided by LongBench, replacing the full context with the surrogate context produced by LC-Boost. For "option [4]", LC-Boost utilizes the prompts provided by LongBench to process each short context and produces the answer as soon as the proper information is found. Table 9 presents an example prompt from LongBench, designed for MultiFieldQA tasks. We modified the prompt by adding the instruction "If no answer can be found in the text, please output "null"". This allows LC-Boost to skip irrelevant short contexts, performing the `[Move]` action. Specifically, for the Code Completion task, LC-Boost reversely browses the context code from near to far as the near context are more useful to predict the code completion. We evaluate all baseline models following the settings provided in LongBench [6]. We use a node with 8 A100 80G GPUs to conduct all experiments.

C Token Consumption Analysis

In Section 3.5, our analysis confirms that LC-Boost significantly reduces energy consumption compared to long LLMs. However, most closed-source LLMs, such as the underlying model of LC-Boost, GPT-3.5-turbo, charge based on token consumption, *e.g.*, US$0.50 per 1M tokens for input and US$1.50 per 1M tokens for output[7]. Consequently, it is crucial to examine whether the

[6] `https://github.com/THUDM/LongBench`

[7] `https://openai.com/api/pricing/`

Table 6: Prompt Template for the `[Task Understanding]` action.

You need to process a task with a long context that greatly exceeds your context limit.
The only feasible way to handle this is by processing the long context chunk by chunk.
Below is the original task prompt:
{task_prompt}
Below is the query:
{input_query}
You have the following options to process the long context. Choose one of them:
[1]. Retrieve the chunk most relevant to the input query to support answer generation.
[2]. Summarize each chunk and then aggregate the summaries after processing all chunks.
[3]. Extract key sentences from each chunk and then aggregate them after processing all chunks.
[4]. Sequentially scan chunks and produce the answer as soon as the query can be answered.
Below are some examples for reference:
The examples begin as follows:
{examples}
The examples conclude here.
Please learn the examples and select one of the options by only outputting the corresponding index number.

Table 7: Query Rewritten Prompt Template for the `[Task Understanding]` action.

You need to process a task with a long context that greatly exceeds your context limit.
The only feasible way to handle this is by processing the long context chunk by chunk.
Below is the original task prompt:
{task_prompt}
Below is the query:
{input_query}
You will process the long context with the following strategy:
{strategy}
Do you think the the query is proper for processing context chunk? If not, rewrite the query.
Below are some examples for reference:
The examples begin as follows:
{examples}
The examples conclude here.
Please study the examples carefully. If the query needs to be rewritten, directly output the revised query.
If no revision is necessary, output "null".

decision-making process of LC-Boost increases token consumption compared to the brute-force method.

To address this issue, we recorded the end-to-end token consumption for three datasets: NarrativeQA, GovReport, and LCC. After token counting, we conclude that LC-Boost's token consumption was 34.1% of the brute-force method's consumption in NarrativeQA, 112% in GovReport, and 29.5% in LCC. These results indicate that LC-Boost's token consumption varies significantly across different tasks. For tasks requiring precise context location, such as QA and code completion, LC-Boost can respond as soon as the relevant context is identified, thereby avoiding the need to process the full context. However, for tasks that necessitate information aggregation, such as summarization, LC-Boost may require more tokens for prompts in each iteration. In practice, for token-consumption-sensitive LLMs, there might be a trade-off between performance and cost-efficiency, which also varies considerably across different tasks.

D Limitations and Broad Impact

In this paper, we propose LC-Boost, a method dedicated to solving long-context tasks using short contexts. However, there are several limitations we would like to address in the future work: (1) Although we conduct comprehensive experiments on many tasks and provide theoretical analysis to support our major claim that most long-context tasks are short-context solvable, there may be more complicated scenarios that require understanding the full context in a brute-force setting. LC-Boost might not be able to process such tasks effectively. (2) As mentioned in Section 2.3, LC-Boost selects actions from a discrete action space. While we argue that the pre-defined action space is versatile

Table 8: Prompt Template for the `[Append]` action.

You are given an article and a question. Read the article carefully and follow my instructions to process it.
Article:
The article begins as follows:
{article}
The article concludes here.
Question:
{question}
Instructions:
Each sentence in the article is marked with a sentence identifier [si], for example [s1].
Select up to ten key sentences from the article that are most likely to answer the question.
Only output the selected sentence identifiers, separated by commas.
Example: [s39],[s54]
If no sentences are relevant, please output "null".

Table 9: Prompt Template for the MultiFieldQA Task from the LongBench Benchmark. Additions made by us are highlighted in blue.

Read the following text and answer briefly.
{context}
Now, answer the following question based on the above text, only give me the answer and do not output any other words. If no answer can be found in the text, please output "null".
Question:{question}
Answer:

Table 10: Prompt Template for the `[Merge]` action.

You are provided with a portion of an article, a question, and summarization of the article's previous portions. Read the article portion and follow my instructions to process it.
Article:
The article begins as follows:
{article}
The article concludes here.
Previous summarization:
The previous summarization is as follows:
{previous_sum}
The previous summarization concludes here.
Question:
{question}
Instruction:
Summarize the partial article to supplement the previous summarization, which can better support the task.
If no content needs to be supplemented, please output "null".

enough to handle most scenarios, a more elegant solution would be to predict actions in a continuous space. We conducted preliminary experiments to explore allowing LC-Boost to prompt itself to predict actions without a predefined action space, such as writing prompts or code autonomously. These experiments resulted in highly unstable performance, particularly for models like GPT-3.5, as such requirements are still challenging. We believe that with a much stronger foundation model, LC-Boost could be expected to predict actions in a continuous space. (3) We choose GPT-3.5 as the foundation model for LC-Boost, instead of open-source LLMs. The reason is that GPT-3.5 is a strong, yet efficient model that can generally understand most instructions. However, we found that most open-source LLMs lack these properties in a zero-shot setting. Fine-tuning these open-source LLMs might be helpful, but constructing such instruction data is infeasible and expensive.

As discussed in Section 3.5, LLMs are likely to become a fundamental infrastructure in the near future. At that scale, their energy consumption will pose significant environmental challenges. As shown in Figure 4, LC-Boost avoids processing long contexts directly by decomposing them into shorter contexts. This approach significantly reduces energy consumption as the context length increases, leading to substantial positive environmental impacts. We believe that in the future, more research will focus on green AI initiatives. This paper could serve as an initial spark to inspire further research in this direction, potentially resulting in broader social impact.