MiniLongBench: The Low-cost Long Context Understanding Benchmark for Large Language Models

Zhongzhan Huang[1], Guoming Ling[1], Shanshan Zhong[1], Hefeng Wu[1], Liang Lin[1,2,3]
[1]Sun Yat-sen University [2]Peng Cheng Laboratory
[3]Guangdong Key Laboratory of Big Data Analysis and Processing

Abstract

Long Context Understanding (LCU) is a critical area for exploration in current large language models (LLMs). However, due to the inherently lengthy nature of long-text data, existing LCU benchmarks for LLMs often result in prohibitively high evaluation costs, like testing time and inference expenses. Through extensive experimentation, we discover that existing LCU benchmarks exhibit significant redundancy, which means the inefficiency in evaluation. In this paper, we propose a concise data compression method tailored for long-text data with sparse information characteristics. By pruning the well-known LCU benchmark LongBench, we create MiniLongBench. This benchmark includes only 237 test samples across six major task categories and 21 distinct tasks. Through empirical analysis of over 60 LLMs, MiniLongBench achieves an average evaluation cost reduced to only 4.5% of the original while maintaining an average rank correlation coefficient of 0.97 with LongBench results. Therefore, our MiniLongBench, as a low-cost benchmark, holds great potential to substantially drive future research into the LCU capabilities of LLMs. See Github for our code, data and tutorial.

1 Introduction

The ability for long context understanding, a.k.a LCU, (Press et al., 2022; Sun et al., 2022; Chen et al., 2023; Zeng et al., 2023; Li et al., 2023a; Beltagy et al., 2020; Roy et al., 2021) is one of the most important areas of exploration for current large language models (LLMs). Tasks with broad applications, such as summarization and question answering based on books, papers, and documents, as well as repository-level code generation, require the capability to handle long context sequences spanning thousands or even tens of thousands of tokens. Currently, the LCU capabilities of LLMs are still in their early stages, and their
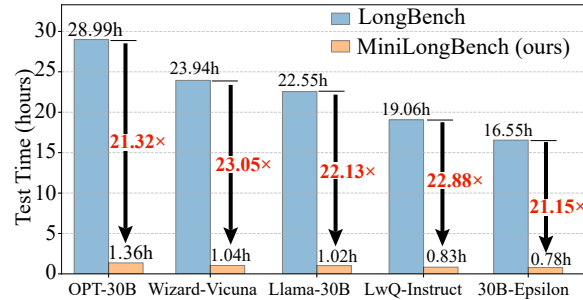

Figure 1: The computational cost of LongBench and MiniLongBench. The proposed MiniLongBench effectively reduces the computational cost of the LongBench, thereby achieving a low-cost LCU benchmark.

rapid development relies on recent proposals of LCU benchmarks (Shaham et al., 2022, 2023; An et al., 2023; Bai et al., 2024d). However, unlike normal LLM benchmarks (Li et al., 2024; Guo et al., 2023; Zhong et al., 2024b, 2023a), LCU benchmarks inherently involve a large number of tokens due to the nature of long context data. Combined with the high number of test samples, the primary challenge these benchmarks face is their high evaluation cost. As shown in Fig. 1, some popular LLMs on 8×RTX3090 GPUs require approximately up to 15 ∼ 30 hours to complete an evaluation on LongBench with one batch size. Moreover, due to the large number of tokens in each long-text data, which significantly increases GPU memory consumption, it is challenging to accelerate testing through multi-batch processing. Therefore, the computational cost illustrated in Fig. 1 cannot be overlooked. Furthermore, when researchers develop new LLM models and need to conduct multiple analyses of LCU capabilities, the time and computational costs become even more prohibitive. Given these challenges, we ask a critical question:

> Do LCU benchmarks really need such a **large** number of test samples?

To answer this question, in this paper, we explore the compression of the well-known LCU bench-
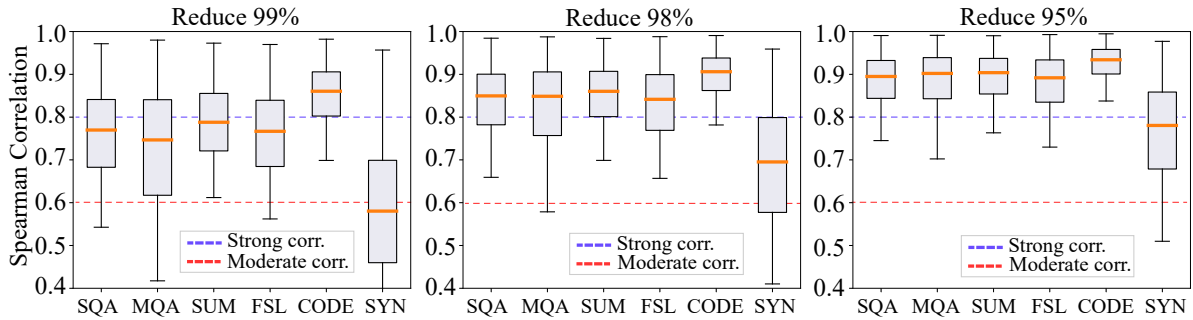
Figure 2: The redundancy of LongBench. "Reduce 95%" means randomly removing 95% of the dataset with equal probability. A Spearman correlation (Sp) ≥ 0.8 indicates a strong correlation and Sp ≥ 0.6 means moderate correlation between the results of randomly sampled subset and LongBench. The abscissa labels from "SQA" to "SYN" represent the abbreviations of the six main tasks in LongBench, with details provided in Section 2.

mark, LongBench (Bai et al., 2024a). In Section 2, we first validate the significant redundancy in the LongBench through a series of random sampling experiments. Furthermore, in Section 3, we propose a simple-yet-effective compression method for long-text data with sparse information, resulting in a compact LCU benchmark, MiniLongBench. Finally, we explore the evaluation results of Mini-LongBench across a range of existing LLMs. Our findings indicate that the proposed MiniLongBench substantially lowers the evaluation cost of LCU capabilities, reducing it to merely 4.5% of the original, while maintaining the assessment outcomes of LLM on LongBench. We show the related works in Appendix D, and summarize the contributions of this paper as follows:

- In this paper, we analyze the redundancy of current LCU benchmark for LLMs and propose an effective method to reduce the number of test samples for low-cost testing.

- Analyzing on over 60 LLMs, our MiniLong-Bench achieves an average ranking correlation of about 0.97 with LongBench while reducing computational cost to only 4.5% of the original.

2 The Redundancy of LCU Benchmark

In this section, we consider the well-known LCU benchmark, LongBench (Bai et al., 2024a), as an example to demonstrate that current LCU benchmarks suffer from significant redundancy. Long-Bench includes nearly 5000 test samples and covers six main task categories, such as single-document question answering (SQA), multi-document question answering (MQA), summarization (SUM), few-shot learning (FSL), code completion (CODE), and synthetic tasks (SYN), which represent key long-text application scenarios. For specific details of LongBench, please refer to the Appendix A.

First, we randomly sample the long-text data from different categories of LongBench n=10,000 times to obtain n subsets of test samples with compression ratio p, where p represents the proportion of remaining samples after sampling. We then test these subsets using dozens of LLMs (see the Appendix B for details), and compute the Spearman correlation (Sp) coefficient (Spearman, 1961) to measure the ranking correlation between the evaluation results of each subset and those of the original LongBench S_L. The closer "Sp" is to 1.0, the more the evaluation of the sampled subset aligns with the evaluation of S_L. We take $p \in \{0.99, 0.98, 0.95\}$ and select the top 7500 results based on Sp for statistical analysis. The experimental results are shown in Fig 2. We find that even though Long-Bench data is randomly reduced by a large amount, some subsets of LongBench still exhibit strong ranking correlations with the original benchmark, e.g., with Sp greater than 0.8 and even 0.9. This indicates that LongBench contains significant redundancy and **does not require so many test samples**. Therefore, in this paper, we will design an efficient method to create a more compact LCU benchmark.

3 Compression for LCU Benchmark

In this section, we explore how to filter long-text data to reduce the size of LCU benchmark, enabling our MiniLongBench for low-cost estimation of LCU capabilities. Although Fig. 2 shows that random sampling can also yield subsets with large Sp, due to its high variance, we need to develop a more stable compression method. A straightforward intuition is that, given a set of m LLMs $\{\ell_i\}_{i=1}^m$, we can leverage their performance on all test samples from LongBench $S_L = \{s_j\}_{j=1}^{|S_L|}$. Using their performance record, we aim to construct a regression model to map these sparsely infor-
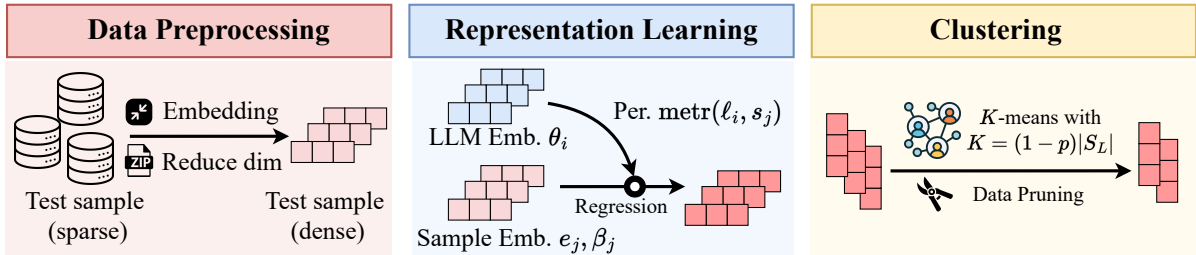
Figure 3: The compression process of the LCU benchmark. "Emb." and "Per." respectively denote embedding and the performance of LLM ℓ_i on sample s_j under the given metric metr$(\cdot, \cdot)$.

Algorithm 1 The construction of MiniLongBench.

Input: The long-text data $S_L = (s_1, s_2, ..., s_{|S_L|})$, reduced dimension d and ratio p. The performance record metr(ℓ_i, s_j) from LLM ℓ_i and sample s_j. Text embedding κ_{test}.
Output: Compact LCU benchmark.

            ▷ Data preprocessing
Intra-sample dimension reduction $s'_j \leftarrow \kappa_{\text{text}}(s_j)$;
Inter-sample dimension reduction by
   $\{e_j\}_{j=1}^{|S_L|} \leftarrow \text{PCA}_d[s'_1, s'_2, ..., s'_{|S_L|}]$;
     ▷ Representation learning for test samples
Initialize LLM ℓ_i's representation by $\theta_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$;
Initialized $\beta_j \leftarrow \mathbf{0}$;
Update learnable (e_j, β_j) and θ_i by Eq. (2);
                   ▷ Clustering
Determine the number of cluster centers $K \leftarrow (1-p)|S_L|$;
Obtain K centers $(c_1, c_2, .., c_K)$ by clustering and (e_j, β_j);
$S_{\text{mini}} \leftarrow (c_1, c_2, .., c_K)$;
return Compact LCU benchmark S_{mini}

---

mative long-text samples into a denser text space first, and gradually project them into a performance space. After that, we can learn the representation of the test samples. Moreover, we then cluster these samples and retain only a certain number of cluster centers as representative test samples, forming S_{mini}, thereby compressing the benchmark. Fig. 3 and Alg. 1 show the compression process of the LCU benchmark and construction of proposed MiniLongBench. Specifically,

(1) **Data Preprocessing**. Unlike data from conventional LLM benchmarks, the effective information in long-text data is highly sparse. Without proper compression of this information, it can significantly impact subsequent representation learning and clustering processes. Therefore, for the sparse information in these long-text data, we initially densify them using a text encoder κ_{text} OpenAIEmbedding (Xian et al., 2024) and a principal component analysis, a.k.a PCA, (Abdi and Williams, 2010) to obtain part of dense $d-$dimensional initialization of test samples, i.e.,

$$\{e_j\}_{j=1}^{|S_L|} = \text{PCA}_d[\{\kappa_{\text{text}}(s_j)\}_{j=1}^{|S_L|}], \quad (1)$$

For a detailed discussion on how data preprocessing influences the construction of MiniLongBench,

please refer to Section 5.

(2) **Representation Learning**. Moreover, similar to (Polo et al., 2024), which utilizes the Item Response Theory in psychology and education (Cai et al., 2016), we can perform representation learning for test samples. Suppose we have LLMs $\{\ell_i\}_{i=1}^m$ and test samples $s_j \in S_L$ with performance measured by the metric metr$(\cdot, \cdot)$. We then assume that the probability of LLM ℓ_i correctly answering sample s_j is given by:

$$\mathbb{P}(\text{metr}(\ell_i, s_j) = 1 | \theta_i, e_j, \beta_j)$$
$$= [1 + \exp(-e_j^\top \theta_i + \beta_j)]^{-1}, \quad (2)$$

where the learnable parameter $\theta_i \in \mathbb{R}^d$ represents the d-dimensional embedding of LLM ℓ_i, initialized using a d-dimensional standard normal distribution. Eq. (2) is classical logistic regression model (Kleinbaum et al., 2002). The parameters (e_j, β_j) are the learnable representations of the test sample s_j, where β_j is initialized to zero vector and e_j initialized by Eq. (1). In this paper, we set $d = 10$. See further analysis on these representations, initialization and d in Section 5.

It is worth noting that in Eq. (2), we use a binary classification example for the metric, where metr$(\ell_i, s_j) = 1$ if ℓ_i performs correctly on s_j, and metr$(\ell_i, s_j) = 0$ otherwise. If metr$(\cdot, \cdot)$ is continuous metrics, it can also be transformed into a binary classification scenario. Specifically, the metric metr$(\cdot, \cdot)$ is generally bounded. For example, in LongBench, metrics such as F1 score, edit distance, etc., are used. We can normalize them to the interval $[0, 1]$, and then consider the following optimization problem.

$$\min_c \| \sum_{i=1}^m \sum_{j=1}^{|S_L|} \text{metr}(\ell_i, s_j)$$
$$- \sum_{i=1}^m \sum_{j=1}^{|S_L|} \mathbf{1}_{[\text{metr}(\ell_i, s_j) \geq c]} \|, \quad (3)$$

Note that the existence of a solution to Eq. (3) is evident. It can be obtained by simply searching

Dataset	Index	Metric	Language	Long. Avg len	MiniLong. Avg len	Long. #data	MiniLong. #data
Single-Document QA							
NarrativeQA	1-1	F1	English	18,409	22,967	200	6 (↓97%)
Qasper	1-2	F1	English	3,619	2,933	200	9 (↓96%)
MultiFieldQA-en	1-3	F1	English	4,559	4,519	150	7 (↓95%)
MultiFieldQA-zh	1-4	F1	Chinese	6,701	6,300	200	15 (↓93%)
Multi-Document QA							
HotpotQA	2-1	F1	English	9,151	8,856	200	13 (↓94%)
2WikiMultihopQA	2-2	F1	English	4,887	4,286	200	13 (↓94%)
MuSiQue	2-3	F1	English	11,214	10,910	200	7 (↓97%)
DuReader	2-4	Rouge-L	Chinese	15,768	12,996	200	6 (↓97%)
Summarization							
GovReport	3-1	Rouge-L	English	8,734	7592	200	12 (↓94%)
QMSum	3-2	Rouge-L	English	10,614	8,253	200	6 (↓97%)
MultiNews	3-3	Rouge-L	English	2,113	1,785	200	11 (↓95%)
VCSUM	3-4	Rouge-L	Chinese	15,380	10,400	200	6 (↓97%)
Few-shot Learning							
TREC	4-1	Acc. (CLS)	English	5,177	6,077	200	8 (↓96%)
TriviaQA	4-2	F1	English	8,209	9,719	200	12 (↓94%)
SAMSum	4-3	Rouge-L	English	6,258	5,974	200	15 (↓93%)
LSHT	4-4	Acc. (CLS)	Chinese	22,337	22,759	200	8 (↓96%)
Synthetic Task							
PassageCount	5-1	Acc. (EM)	English	11,414	10,627	200	4 (↓98%)
PassageRetrieval-en	5-2	Acc. (EM)	English	9,289	9,394	200	15 (↓93%)
PassageRetrieval-zh	5-3	Acc. (EM)	Chinese	6,745	6,684	200	15 (↓93%)
Code Completion							
LCC	6-1	Edit Sim	Python/C#/Java	1,235	1,187	500	26 (↓95%)
RepoBench-P	6-2	Edit Sim	Python/Jave	4,206	3,723	500	23 (↓95%)

Table 1: The dataset statistics in LongBench and MiniLongBench. "Long." and "MiniLong." denote LongBench and MiniLongBench. "Avg len" (average length) is computed using the number of words for the English (code) datasets and the number of characters for the Chinese datasets. "Acc. (CLS)" refers to classification accuracy, while "Acc. (EM)" refers to exact match accuracy. "#data" means the number of data.
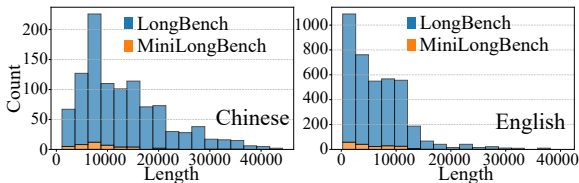

Figure 4: The Length distribution for English and Chinese data in LongBench and MiniLongBench, measured by the number of words and characters.

the interval $[0, 1]$ to get an approximate solution for c. Once c is obtained, we replace the original $\mathrm{metr}(\ell_i, s_j)$ with $\mathrm{metr}(\ell_i, s_j)' = \mathbf{1}_{[\mathrm{metr}(\ell_i,s_j) \geq c]}$ which can transform the continuous metric into a discrete binary scenario similar to Eq. (2).

(3) **Clustering**. Next, we update θ_i, e_j, and β_j simultaneously using the training approach of logistic regression. Once these learnable parameters converge, we concatenate (e_j, β_j) as the final representation of the test sample s_j, and perform clustering analysis on them using K-Means (Hamerly and Elkan, 2003) under Euclidean distance, where $K = (1-p)|S_L|$. Finally, the all cluster centers are integrated as the test samples of MiniLongBench

S_{mini}. In Section 4, we will further validate the effectiveness of MiniLongBench from an experimental perspective.

4 Compact LCU Benchmark: MiniLongBench

In this section, we present our compact MiniLongBench and demonstrate through comprehensive experiments that it significantly reduces computational costs while preserving original LongBench's evaluation effectiveness. We select over 60 LLMs for analysis, with $m = 20$ of them participating in the training described in Section 3, and the rest serving as candidates for validating the effectiveness of MiniLongBench. See Appendix B for details of LLMs considered.

4.1 The Details of MiniLongBench

Chosing compression ratio $p = 0.95$, we use the compression method shown in Section 3 for Long-Bench to obtain compact LCU benchmark Mini-LongBench. This benchmark includes only 237 test samples across six task categories, with an average length of 6193 words (English) and 10344

Model	SQA	MQA	SUM	FSL	SYN	CODE
DeepSeek-V3-128k	0.43	0.31	0.12	0.67	0.26	0.88
GPT-4o-mini-128k	0.41	0.30	0.10	0.65	0.23	0.87
GPT-3.5-Turbo-16k	0.37	0.25	0.07	0.58	0.17	0.81
Internlm3-8B-32k	0.33	0.24	0.04	0.53	0.12	0.64
ChatGLM3-6B-8k	0.17	0.08	0.02	0.39	0.02	0.48
ChatGLM4-9B-128k	0.36	0.27	0.06	0.58	0.14	0.75
Qwen-7B-8k	0.20	0.10	0.04	0.48	0.07	0.71
Qwen2-7B-128k	0.29	0.22	0.05	0.55	0.09	0.71
Qwen2.5-7B-128k	0.34	0.24	0.04	0.55	0.12	0.67
Qwen2.5-14B-128k	0.35	0.26	0.05	0.56	0.12	0.66
Qwen2.5-32B-128k	0.36	0.26	0.06	0.59	0.15	0.74
Llama-7B-2k	0.09	0.04	0.02	0.39	0.02	0.61
Llama2-7B-4k	0.11	0.04	0.02	0.41	0.04	0.66
Llama3-8B-8k	0.11	0.03	0.04	0.47	0.08	0.71
Llama-30B-2k	0.10	0.04	0.02	0.41	0.04	0.65
OPT-30B-2k	0.08	0.03	0.01	0.33	0.02	0.48
Wizard-Vicuna-2k	0.20	0.13	0.02	0.41	0.05	0.59
LwQ-Instruct-2k	0.23	0.18	0.04	0.45	0.07	0.70
30B-Epsilon-2k	0.20	0.13	0.04	0.48	0.07	0.73

Table 2: Specific evaluation results on MiniLongBench. See Appendix C and Appendix G for the more analysis and detail results on various advanced LLMs.

characters (Chinese). Consistent with LongBench, MiniLongBench has six major task categories and 21 distinct tasks, covering key long-text application scenarios. Through the long-text dataset compression method proposed in Alg. 1, these different tasks have been compressed by about 95%, greatly reducing the computational consumption of the LCU benchmark in the testing process. The specific statistics is shown in Table 1.

As shown in Table 1, the average length of Mini-LongBench is smaller compared to that of Long-Bench due to data pruning, but it generally maintains a similar magnitude. This indicates that Long-Bench retains a good diversity of long-text data even after compression. Moreover, we further illustrate the length distribution of data across different languages, including English and Chinese, in Fig. 4. We observe that for different languages, our proposed MiniLongBench significantly reduces the total length of data input to the LLM, thereby greatly decreasing the number of tokens in the model input and reducing computational costs. The further discussions with other compression ratio p and m trained LLMs are shown in Section 5.

4.2 The Evaluation Method

In this section, we explore how to evaluate the LCU capabilities of LLMs using MiniLongBench. A straightforward approach is to directly assess them on MiniLongBench, yielding reliable results with a Sp of 0.95 compared to LongBench (see Appendix G for details). However, it's important to note that MiniLongBench, having significantly
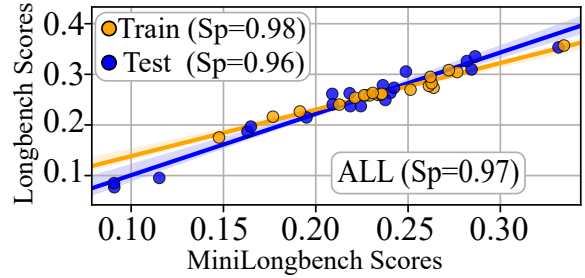

Figure 5: The analysis of rank correlation (Sp) between LongBench and MiniLongBench.

fewer test samples than LongBench, may introduce some evaluation bias. To mitigate this, we can use MiniLongBench samples to estimate the performance (Polo et al., 2024; Pacchiardi et al., 2024) of the LLMs on LongBench, thereby reducing bias and achieving an improved Sp of up to 0.97.

Specifically, For a new LLM ℓ_0 to be tested, we first evaluate it on all test samples c_j from MiniLongBench S_{mini} and obtain its performance $\text{metr}(\ell_0, c_j)$. Subsequently, we apply consistent normalization and discretization for $\text{metr}(\ell_0, c_j)$ as outlined in Section 3, and initialize a d-dimensional feature vector $\bar{\theta}$ for the LLM ℓ_0 using a standard normal distribution.

Next, we fine-tune $\bar{\theta}$ on the test samples of S_{mini} using Eq. (2) to adapt it to the representation space of the test samples. After completing the fine-tuning, we can construct the following MiniLong-Bench score through Eq. (2) to estimate the performance of ℓ_0 across the entire S_L:

$$\sum_{j=1}^{|S_L|}[1 + \exp(-e_j\top\bar{\theta} + \beta_j)]^{-1}/|S_L|, \quad (4)$$

The time required for fine-tuning in the aforementioned evaluation process and the storage cost for the features of S_L are both minimal, requiring only about 10 MB of disk space and as little as 0.03 seconds of GPU time, even on a laptop. For specific statistics, please refer to Appendix H.

4.3 The Evaluation Results

Moreover, Fig. 5 shows the rank correlation between LongBench and the proposed MiniLong-Bench are 0.96~0.98, whether on the LLMs that participated in the training or on other unseen LLMs. Moreover, in conjunction with the results presented in Fig.1, this indicates that the proposed MiniLongBench can effectively replicate the evaluation outcomes of LongBench while maintaining very low computational costs.

Additionally, we present in Table 2 the specific performance of various advanced LLMs across dif-
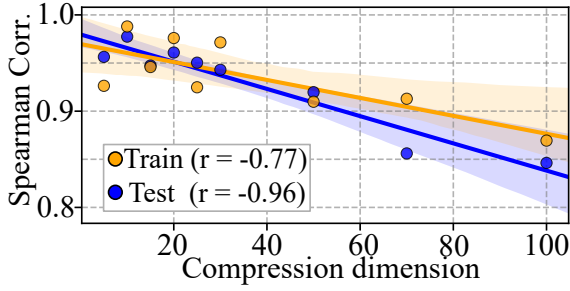
Figure 6: The impact of compression dimension d on the construction of MiniLongBench. "r" is Pearson correlation coefficient.

Figure 7: Further analysis for MiniLongBench. The results of (a) various compression ratio p, (b) various text embedding κ_{text}. (c) The influence of various β_j. The bars with darker color represent the settings adopted by our settings. "rand" and "randn" denote the standard uniform and normal distribution. "Longf." and "Open." are Longformer and OpenAIEmbedding.

ferent tasks on the proposed MiniLongBench. For more detailed results, please refer to Appendix C.

5 Analysis

In this Section, We conduct a more comprehensive analysis of the proposed MiniLongBench.

(1) How does the reduced dimension d affect the compression of the LCU benchmark?

In Eq. (1) of Session 3, we perform initial compression of the long-text data in LongBench using text embedding OpenAIEmbedding (Xian et al., 2024) and a PCA (Abdi and Williams, 2010), allowing the long-text information to be initialized as some vectors with dimension d.

In this section, we further explore the specific impact of the compressed dimension d on constructing a compact MiniLongBench. Specifically, following the experiment setting in Section 4, we consider $d \in \{5, 10, 15, 20, 25, 30, 50, 70, 100\}$ and present the Sp of the evaluation results for Long-Bench and MiniLongBench under different values of d in Fig. 6. We observe that a negative correlation between Sp and d. This indicates that for long-texts data with sparse information, using excessively high-dimensional representations is not advisable, as it can still lead to sparse representations even after representation learning. Further information compression is crucial. In this paper, we set $d = 10$ by default.

(2) Is PCA necessary for MiniLongBench?

In data preprocessing, PCA is employed to further reduce the dimensionality of features after text embedding. If we remove the PCA operation in Eq. (1), on one hand, the high dimension of κ_{text}, which reaches 1024, would result in significant additional computational overhead during training.
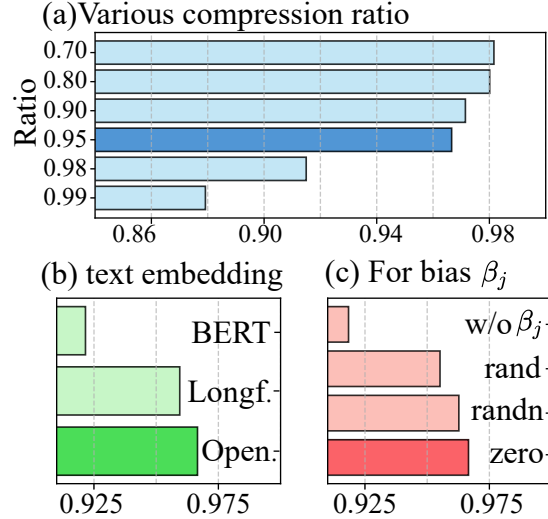
Moreover, due to the large dimensionality, we observe that the average Sp of MiniLongBench and LongBench drops from 0.95 to 0.67, which aligns with the phenomenon observed in Fig. 6. Therefore, the dimension reducing method, like PCA, is essential for the construction of MiniLongBench.

(3) How to select the text embedding κ_{text}.

In the data preprocessing phase, we utilize OpenAIEmbedding (Xian et al., 2024) for text embedding. In Fig. 7 (b), we present the results of employing alternative text embeddings, including Longformer (Zhu et al., 2021) and BERT (Liu et al., 2019). We observe that BERT, which only supports token inputs with a maximum length of 512, significantly underperforms compared to OpenAIEmbedding and Longformer, which support lengths of 8192 and 4096, respectively. This is primarily due to BERT's weaker capability in information densification and the inevitable information loss when handling test samples exceeding the token length limit, as they can only be processed through chunked densification. Therefore, this paper defaults to using the more capable OpenAIEmbedding.

(4) How about other compression ratios p?

In this paper, we set the compression ratio $p = 0.95$ as the default. Subsequently, we further explore the selection of p in Fig. 7 (a). We observe that as p ap-
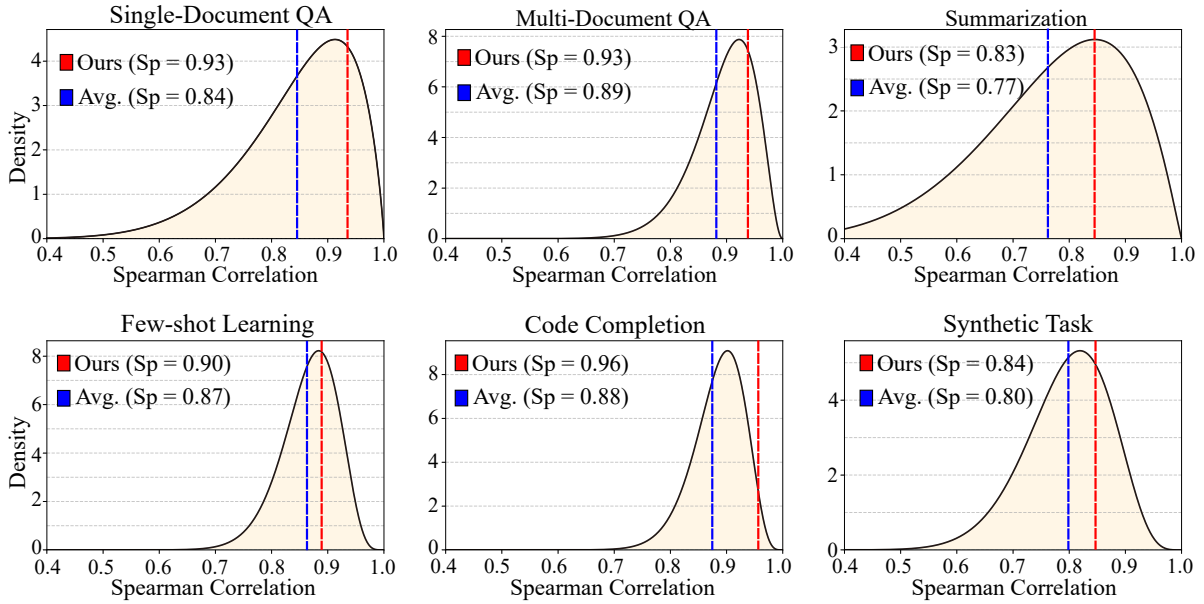
Figure 8: The impact of the selection of LLMs on the construction of MiniLongBench .

proaches 1, meaning more test samples are reduced, the Sp between MiniLongBench and LongBench decreases, which aligns with the observations in Fig. 2. This is because, although the LCU benchmark has significant redundancy in test samples, an extremely low compression ratio can easily disrupt the data distribution or diversity of the benchmark, leading to substantial bias in the evaluation of LLMs. Based on the experimental results in Fig. 7 (a), $p = 0.95$ is a favorable choice, as it balances both the testing cost and the evaluation capability of the benchmark.

(5) Is the learnable bias β_j important?

In Eq. (2), we introduce a learnable bias β_j for the logistic regression model. In Fig. 7 (c), we explore its impact on the construction of MiniLongBench by testing different initializations and removing it entirely. We observe that, on one hand, the inclusion of β_j aids in the representation learning of test samples, as removing it results in a noticeable decline in Sp. On the other hand, different initializations yield varying performance levels, with zero initialization achieving the best results. In conclusion, the setting of learnable β_j is important, and we employ a learnable bias with zero initialization.

(6) About the selection of m LLMs.

In this section, we further analyze the impact of the LLMs involved in training on the construction of MiniLongBench from the selection of LLMs.

We fix the number of LLMs, m, and then independently sample 1000 times from all the LLMs considered in this paper. Using the method mentioned in Section 3, we obtain various compact new "MiniLongBench" and compute its Sp distribution against LongBench evaluation results. The results are shown in Fig. 8. We find that the choice of LLMs involved in training significantly affects the construction of MiniLongBench, which is intuitive. This is because the representation of test samples depends on the performance records of the LLMs on LongBench, and when the selected LLMs perform poorly, their representations struggle to correctly project the test samples into the performance space. In this paper, we manually select a few LLMs with generally good performance across various aspects to participate in the construction of MiniLongBench. A list of the chosen LLMs can be found in Appendix B. In the future, the automated LLMs selection is needed.

(7) What is the appropriate number of LLMs m to involve in training?

Moreover, we further explore the impact of the number of LLMs involved in training. For a specific number of LLMs, m, we repeat the independent sampling 5 times and compute the average Sp of the constructed MiniLongBench and LongBench evaluation results across all LLMs. The results are shown in Fig. 9. We observe that as m increases, Sp gradually increases and approaches 1.0. This indicates that involving enough LLMs is beneficial for the representation learning of test samples. And we also note that when $m = 20$, the Sp in different tasks seems acceptable, suggesting that
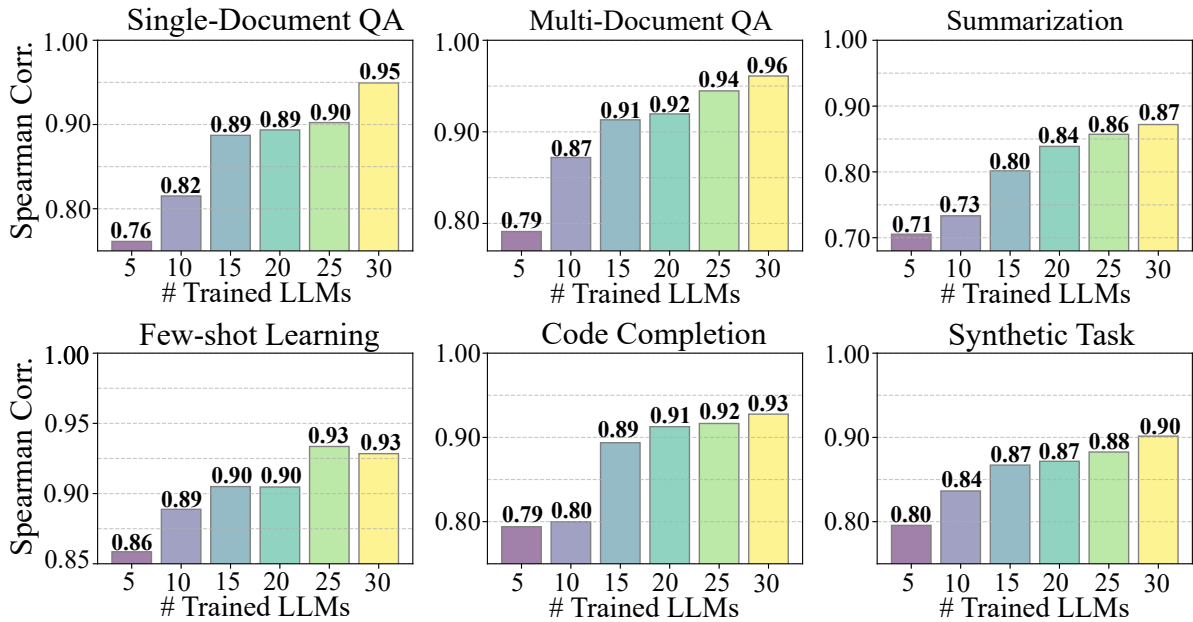
11448

Figure 9: The impact of the number of LLMs m on the construction of MiniLongBench .

although the number of LLMs aids in representation learning, there is still considerable redundancy. Considering the computational cost, we take the acceptable $m = 20$ as default.

---

(8) Is the average Sp $\geq$ 0.97 enough?

In Section 4, we show that the proposed MiniLong-Bench achieves an average Sp of 0.97 compared to LongBench. And, we also find that the p-value is less than 0.001, indicating the ranking correlation is not only very strong but also highly statistically significant. Next, It is noted that since Sp cannot completely reach 1.0, therefore, the errors are inevitably present.

To demonstrate the usability of MiniLongBench with Sp = 0.97, in addition to the experiment in Fig. 5, we consider directly visualizing the ranking results of different LCU benchmarks. As shown in Fig. 10, for some random test samples, we also randomly selected 8 different LLMs to compare their ranking results on MiniLongBench and LongBench. We can observe that the ranking results are quite similar, despite some minor discrepancies. For more results, please refer to Appendix F. In the future, we should further refine the compression methods to bring Sp as close as possible to 1.0 across all subtasks.

---

(9) Why not just random sampling?

In Fig. 2, we show that through random sampling, we identify a significant amount of redundancy in LongBench. However, relying solely on random
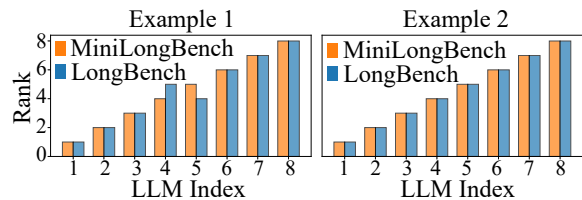
Figure 10: The visualization of ranking. See more ranking examples in Appendix F.

sampling to compress LongBench is insufficient. The primary reason is that while random sampling can probabilistically yield high Sp results, as shown in Fig. 2, the variance is substantial, making it easy to achieve suboptimal compression.

The compression method we propose in Section 3 for the LCU benchmark effectively mitigates these issues, consistently achieving high Sp across various subtasks.

6 Conclusion

In this paper, we propose a concise data compression method for long-text data with sparse information. By pruning the well-known LCU benchmark LongBench, we created MiniLongBench. Through empirical analysis of over 60 LLMs with varying performance levels, MiniLongBench achieved an average evaluation cost reduction to 4.5% of the original, while maintaining strong consistency with LongBench results. This phenomenon indicates that the proposed MiniLongBench has great potential to greatly promote the exploration of LLMs' LCU capabilities in the future.

Limitations

The LCU benchmark compression method shown in this paper requires performance records from various LLMs as training data. However, most of this data is not open-source in practice. Consequently, we need to incur significant API costs and GPU computational resources to obtain this data. On the other hand, although we have achieved effective compression for LongBench, since Sp cannot be 1.0, we cannot expect MiniLongBench to have exactly the same evaluation capabilities as LongBench, only nearly identical. Additionally, there is still considerable room for performance improvement in the summarization and synthetic tasks, which are worthwhile directions for future enhancements.

Acknowledgments

References

Herve Abdi and Lynne J Williams. 2010. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459.

Rishabh Agarwal, Avi Singh, Lei M Zhang, Bernd Bohnet, Luis Rosias, Stephanie Chan, Biao Zhang, Ankesh Anand, Zaheer Abbas, Azade Nova, et al. 2024. Many-shot in-context learning. *arXiv preprint arXiv:2404.11018*.

Joshua Ainslie, Tao Lei, Michiel de Jong, Santiago Ontañón, Siddhartha Brahma, Yury Zemlyanskiy, David Uthus, Mandy Guo, James Lee-Thorp, Yi Tay, et al. 2023. Colt5: Faster long-range transformers with conditional computation. *arXiv preprint arXiv:2303.09752*.

Chenxin An, Shansan Gong, Ming Zhong, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. L-eval: Instituting standardized evaluation for long context language models. *arXiv preprint arXiv:2307.11088*.

Chenxin An, Shansan Gong, Ming Zhong, Xingjian Zhao, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2024. L-eval: Instituting standardized evaluation for long context language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14388–14411, Bangkok, Thailand. Association for Computational Linguistics.

Anthropic. 2024. Anthropic: Introducing claude 3.5 sonnet.

Yushi Bai, Xin Lv, and et al. 2024a. Longbench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 3119–3137. Association for Computational Linguistics.

Yushi Bai, Xin Lv, Jiajie Zhang, Yuze He, Ji Qi, Lei Hou, Jie Tang, Yuxiao Dong, and Juanzi Li. 2024b. LongAlign: A recipe for long context alignment of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1376–1395, Miami, Florida, USA. Association for Computational Linguistics.

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. 2024c. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, et al. 2024d. Longbench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. *arXiv preprint arXiv:2412.15204*.

Yushi Bai, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024e. Longwriter: Unleashing 10,000+ word generation from long context llms. *arXiv preprint arXiv:2408.07055*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

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

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. 2022. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091.

Li Cai, Kilchan Choi, Mark Hansen, and Lauren Harrell. 2016. Item response theory. *Annual Review of Statistics and Its Application*, 3(1):297–321.

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

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner. 2021. 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.

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

Zican Dong, Tianyi Tang, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. 2024. Bamboo: A comprehensive benchmark for evaluating long text modeling capacities of large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2086–2099.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Alexander Richard Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270.

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. 2024. Data engineering for scaling language models to 128K context. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 14125–14134. PMLR.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. 2024. Datacomp: In search of the next generation of multimodal datasets. *Advances in Neural Information Processing Systems*, 36.

Tianyu Gao, Alexander Wettig, Howard Yen, and Danqi Chen. 2024. How to train long-context language models (effectively). *arXiv preprint arXiv:2410.02660*.

Team GLM, Aohan Zeng, Bin Xu, and et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

Taicheng Guo, Bozhao Nan, Zhenwen Liang, Zhichun Guo, Nitesh Chawla, Olaf Wiest, Xiangliang Zhang, et al. 2023. What can large language models do in chemistry? a comprehensive benchmark on eight tasks. *Advances in Neural Information Processing Systems*, 36:59662–59688.

Greg Hamerly and Charles Elkan. 2003. Learning the k in k-means. *Advances in neural information processing systems*, 16.

Wei He, Zhongzhan Huang, Mingfu Liang, Senwei Liang, and Haizhao Yang. 2021. Blending pruning criteria for convolutional neural networks. In *Artificial Neural Networks and Machine Learning–ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part IV 30*, pages 3–15. Springer.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021a. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436.

Zhongzhan Huang, Senwei Liang, Mingfu Liang, Wei He, Haizhao Yang, and Liang Lin. 2022. The lottery ticket hypothesis for self-attention in convolutional neural network. *arXiv preprint arXiv:2207.07858*.

Zhongzhan Huang, Wenqi Shao, Xinjiang Wang, Liang Lin, and Ping Luo. 2021b. Rethinking the pruning criteria for convolutional neural network. *Advances in Neural Information Processing Systems*, 34:16305–16318.

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

Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. 2024. Shortened llama: A simple depth pruning for large language models. *arXiv preprint arXiv:2402.02834*, 11.

Alex Kipnis, Konstantinos Voudouris, Luca M Schulze Buschoff, and Eric Schulz. 2024. metabench–a sparse benchmark to measure general ability in large language models. *arXiv preprint arXiv:2407.12844*.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. 2002. *Logistic regression*. Springer.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. 2024. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. *arXiv preprint arXiv:2409.12941*.

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

Philippe Laban, Alexander Richard Fabbri, Caiming Xiong, and Chien-Sheng Wu. 2024. Summary of a haystack: A challenge to long-context llms and rag systems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9885–9903.

Shiye Lei and Dacheng Tao. 2023. A comprehensive survey of dataset distillation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Bohao Li, Yuying Ge, Yixiao Ge, Guangzhi Wang, Rui Wang, Ruimao Zhang, and Ying Shan. 2024. Seed-bench: Benchmarking multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13299–13308.

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

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

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Senwei Liang, Zhongzhan Huang, Mingfu Liang, and Haizhao Yang. 2020. Instance enhancement batch normalization: An adaptive regulator of batch noise. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 4819–4827.

Xinnian Liang, Bing Wang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. 2023. Unleashing infinite-length input capacity for large-scale language models with self-controlled memory system. *arXiv preprint arXiv:2304.13343*.

Haokun Lin, Haoli Bai, Zhili Liu, Lu Hou, Muyi Sun, Linqi Song, Ying Wei, and Zhenan Sun. 2024. Mopeclip: Structured pruning for efficient vision-language models with module-wise pruning error metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27370–27380.

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

Xiang Liu, Peijie Dong, Xuming Hu, and Xiaowen Chu. 2024. Longgenbench: Long-context generation benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 865–883.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*.

Pedro Henrique Martins, Zita Marinho, and André FT Martins. 2022. ∞-former: Infinite memory transformer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5468–5485.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Bhuminand Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. 2024. Compact language models via pruning and knowledge distillation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

OpenAI. 2024. Openai: Hello gpt-4o.

Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. 2023. Resurrecting recurrent neural networks for long sequences. *arXiv preprint arXiv:2303.06349*.

Lorenzo Pacchiardi, Lucy G Cheke, and José Hernández-Orallo. 2024. 100 instances is all you need: predicting the success of a new llm on unseen data by testing on a few instances. *arXiv preprint arXiv:2409.03563*.

Richard Yuanzhe Pang, Alicia Parrish, Nitish Joshi, Nikita Nangia, Jason Phang, Angelica Chen, Vishakh Padmakumar, Johnny Ma, Jana Thompson, He He, et al. 2022. 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*.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. 2024. tinybenchmarks: evaluating llms with fewer examples. In *Forty-first International Conference on Machine Learning*.

Ofir Press, Noah Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*.

Haoran Que, Feiyu Duan, Liqun He, Yutao Mou, Wangchunshu Zhou, Jiaheng Liu, Wenge Rong, Zekun Moore Wang, Jian Yang, Ge Zhang, et al. 2024. Hellobench: Evaluating long text generation capabilities of large language models. *arXiv preprint arXiv:2409.16191*.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*.

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

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2021. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68.

Noveen Sachdeva and Julian McAuley. 2023. Data distillation: A survey. *arXiv preprint arXiv:2301.04272*.

Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. 2023. Zeroscrolls: A zero-shot benchmark for long text understanding. *arXiv preprint arXiv:2305.14196*.

Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, et al. 2022. Scrolls: Standardized comparison over long language sequences. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 12007–12021.

Mingyang Song, Mao Zheng, and Xuan Luo. 2024. Counting-stars: A simple, efficient, and reasonable strategy for evaluating long-context large language models. *arXiv preprint arXiv:2403.11802*.

Charles Spearman. 1961. The proof and measurement of association between two things.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2022. A length-extrapolatable transformer. *arXiv preprint arXiv:2212.10554*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2022. Efficient transformers: A survey. *ACM Comput. Surv.*, 55(6).

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers and distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR.

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

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Kiran Vodrahalli, Santiago Ontanon, Nilesh Tripuraneni, Kelvin Xu, Sanil Jain, Rakesh Shivanna, Jeffrey Hui, Nishanth Dikkala, Mehran Kazemi, Bahare Fatemi, et al. 2024. Michelangelo: Long context evaluations beyond haystacks via latent structure queries. *arXiv preprint arXiv:2409.12640*.

Alex Wang, Richard Yuanzhe Pang, Angelica Chen, Jason Phang, and Samuel Bowman. 2022. Squality: Building a long-document summarization dataset the hard way. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1139–1156.

Minzheng Wang, Longze Chen, Fu Cheng, Shengyi Liao, Xinghua Zhang, Bingli Wu, Haiyang Yu, Nan Xu, Lei Zhang, Run Luo, et al. 2024. Leave no document behind: Benchmarking long-context llms with extended multi-doc qa. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5627–5646.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Yuhao Wu, Ming Shan Hee, Zhiqing Hu, and Roy Ka-Wei Lee. 2024. Longgenbench: Benchmarking long-form generation in long context llms. *arXiv preprint arXiv:2409.02076*.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. Memorizing transformers. In *International Conference on Learning Representations*.

Jasper Xian, Tommaso Teofili, Ronak Pradeep, and Jimmy Lin. 2024. Vector search with openai embeddings: Lucene is all you need. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 1090–1093.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. 2024. Effective long-context scaling of foundation models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4643–4663.

Yifei Yang, Zouying Cao, and Hai Zhao. 2024. Laco: Large language model pruning via layer collapse. *arXiv preprint arXiv:2402.11187*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Howard Yen, Tianyu Gao, Minmin Hou, Ke Ding, Daniel Fleischer, Peter Izsak, Moshe Wasserblat, and Danqi Chen. 2024. Helmet: How to evaluate long-context language models effectively and thoroughly. *arXiv preprint arXiv:2410.02694*.

Ruonan Yu, Songhua Liu, and Xinchao Wang. 2023. Dataset distillation: A comprehensive review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2023. Glm-130b: An open bilingual pre-trained model. In *The Eleventh International Conference on Learning Representations*.

Jiajie Zhang, Yushi Bai, Xin Lv, Wanjun Gu, Danqing Liu, Minhao Zou, Shulin Cao, Lei Hou, Yuxiao Dong, Ling Feng, et al. 2024a. Longcite: Enabling llms to generate fine-grained citations in long-context qa. *arXiv preprint arXiv:2409.02897*.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Hao, Xu Han, Zhen Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024b. ∞Bench: Extending long context evaluation beyond 100K tokens. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15262–15277, Bangkok, Thailand. Association for Computational Linguistics.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, et al. 2021. Qmsum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921.

Shanshan Zhong, Shanghua Gao, Zhongzhan Huang, Wushao Wen, Marinka Žitnik, and Pan Zhou. 2024a. Moextend: Tuning new experts for modality and task extension. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 80–91.

Shanshan Zhong, Zhongzhan Huang, Shanghua Gao, Wushao Wen, Liang Lin, Marinka Zitnik, and Pan Zhou. 2024b. Let's think outside the box: Exploring leap-of-thought in large language models with creative humor generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13246–13257.

Shanshan Zhong, Zhongzhan Huang, Weushao Wen, Jinghui Qin, and Liang Lin. 2023a. Sur-adapter: Enhancing text-to-image pre-trained diffusion models with large language models. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 567–578.

Shanshan Zhong, Jinghui Qin, Zhongzhan Huang, and Daifeng Li. 2022. Cem: Machine-human chatting handoff via causal-enhance module. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3242–3253.

Shanshan Zhong, Wushao Wen, Jinghui Qin, Qiangpu Chen, and Zhongzhan Huang. 2023b. Lsas: Lightweight sub-attention strategy for alleviating attention bias problem. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pages 2051–2056. IEEE.

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

Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. 2021. Long-short transformer: Efficient transformers for language and vision. *Advances in neural information processing systems*, 34:17723–17736.

A The Details of LongBench

LongBench (Bai et al., 2024a) represents the first bilingual, multi-task benchmark specifically developed for assessing long-context comprehension. The benchmark encompasses six primary task categories and 21 distinct tasks, spanning crucial long-text application domains (Dasigi et al., 2021; Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022; Huang et al., 2021a; Zhong et al., 2021; Fabbri et al., 2019; Ainslie et al., 2023; Li and Roth, 2002) including multi-document QA, single-document QA, summarization, few-shot learning, code completion, and synthetic tasks, as detailed in Table 1.

To thoroughly evaluate large models' bilingual proficiency in long-context processing, LongBench incorporates tasks in both Chinese and English. The dataset comprises 4,750 test instances, with average lengths of 6,711 words and 13,386 characters for English and Chinese respectively, ensuring extensive coverage of diverse scenarios. The challenge of long-context understanding (Press et al., 2022; Sun et al., 2022; Chen et al., 2023; Zhong et al., 2024a, 2022) can be formally defined as follows: given an input sequence I and a context sequence C, the model is tasked with generating an output A. For example, in a QA task, I represents the question, C corresponds to the document, and A is the answer. Across LongBench, I and A are typically short, whereas C can span thousands of tokens. Specific instantiations of (I, C, A) for each task are provided in Table 7 of (Bai et al., 2024a).

B The LLMs Considered in MiniLongBench

In this section, we list all LLMs we considered in Table 3. Among them, 20 LLMs were utilized for training to aid in obtaining effective representations of test samples in the LCU benchmark. In this study, we have carefully curated a selection of LLMs that demonstrate consistently strong performance across multiple dimensions to contribute to the development of MiniLongBench. These models were chosen based on their proven capabilities in various tasks and benchmarks. However, to enhance the scalability and objectivity of our approach, future work should focus on implementing an automated LLM selection mechanism. This advancement would not only streamline the selection process but also ensure a more systematic and unbiased evaluation of potential models for inclusion in MiniLongBench.

Model	Type	Model	Type
ALMA-7B-Ja-V2	T	Amd-llama-135m	A
GOAT-7B-Community	T	Amd-llama-135m-code	A
Koss-7B-chat	T	Distilled-HermesChat-7B	A
Kunoichi-7B	T	Loyal-Macaroni-Maid-7B	A
Llama-2-7b-ft-instruct-es	T	Llama-3-8b-hf	A
Llama-2-7b-hf	T	Llama-7b-SFT_ds_wiki65k	A
Llama-7b-hf	T	Llama-shishya-7b-ep3-v1	A
Mistral-7B-Instruct-v0.2	T	Llama-30b	A
OLMo-1B	T	OLMo-1B-SFT	A
Airoboros-7b	T	StopCarbon-10.7B-v6	A
Gemma2-9b-hf	T	Synatra-RP-Orca-2-7b-v0.1	A
Giraffe-7b	T	TowerInstruct-7B-v0.1	A
Mistral-7b-v0.1-hf	T	Gemma2-2b-hf	A
Perry-7b	T	Manatee-7b	A
Qwen-7b-hf	T	Mistral-7b-v0.3-hf	A
Qwen1.5-0.5b-hf	T	Qwen1.5-1.8b-hf	A
Qwen2-0.5b-hf	T	Qwen2.5-0.5b-instruct-hf	A
Qwen2.5-0.5b-base	T	Qwen2.5-3b-base	A
Qwen2.5-1.5b-base	T	Qwen2.5-3b-instruct-hf	A
Tulu-7B-fp16	T	Recycled-wizardlm-7b-v2.0	A
OPT-30B	A	Wizaed-Vicuna	A
Llama-30B	A	LwQ-Instruct	A
30B-Epsilon	A	DeepSeek V3	A
GPT o1 mini	A	GPT-3.5-turbo	A
ChatGLM-6B	A	ChatGLM3-6B	A
ChatGLM3-9B	A	Qwen-7B	A
Qwen2-7B	A	Qwen2.5-7B	A
Qwen2.5-14B	A	Qwen2.5-32B	A
Gemma2-9B	A	Llama-7B-2k	A
Llama2-7B-4k	A	Llama3-8B	A

Table 3: The LLMs considered in MiniLongBench. "T" and "A" denote "for tranining" and "for analysis".

C The Details Results of Advanced LLMs

In Section 4.3, we present the direct performance results of some Advanced LLMs on the six main tasks of MiniLongBench. Note that MiniLongBench includes not only the six main tasks but also 21 subtasks. Therefore, in this section, we will display the detailed results. The results are shown in Table 4 and Table 5, where the indices in the table correspond to those in Table 2 in the main text.

In addition to the performance estimation of the target LLM on the entire LongBench using MiniLongBench's test samples, as demonstrated in Section 4.3, we further propose a more straightforward but slightly less effective method for evaluating LCU capabilities in Appendix G. Specifically, the target LLM is directly tested on MiniLongBench's test samples without requiring any additional steps.

D Related Works

D.1 Long Context Understanding (LCU)

Existing research on LCU in LLMs primarily addresses two critical challenges in long-text modeling: the substantial runtime overhead associated with extended contexts and the issue of catastrophic forgetting during long sequence processing.

A significant body of work has concentrated on

Model	Single-Doc QA					Multi-Doc QA					Summarization				
	1-1	1-2	1-3	1-4	Avg	2-1	2-2	2-3	2-4	Avg	3-1	3-2	3-3	3-4	Avg
DeepSeek-V3-128k	0.29	0.43	0.50	0.52	0.43	0.51	0.45	0.21	0.10	0.31	0.24	0.08	0.16	0.01	0.12
GPT-4o-mini-128k	0.27	0.41	0.49	0.48	0.41	0.48	0.42	0.19	0.09	0.30	0.19	0.07	0.15	0.01	0.10
GPT-3.5-Turbo-16k	0.21	0.36	0.47	0.43	0.37	0.42	0.37	0.15	0.07	0.25	0.13	0.04	0.10	0.01	0.07
Internlm3-8B-32k	0.17	0.32	0.43	0.40	0.33	0.40	0.39	0.14	0.05	0.24	0.07	0.02	0.06	0.01	0.04
ChatGLM3-6B-8k	0.05	0.14	0.28	0.21	0.17	0.12	0.14	0.03	0.03	0.08	0.03	0.01	0.03	0.01	0.02
ChatGLM4-9B-128k	0.21	0.36	0.44	0.45	0.36	0.44	0.42	0.17	0.06	0.27	0.10	0.04	0.09	0.01	0.06
Qwen-7B-8k	0.08	0.18	0.30	0.24	0.20	0.17	0.15	0.05	0.04	0.10	0.07	0.02	0.06	0.00	0.04
Qwen2-7B-128k	0.15	0.30	0.38	0.34	0.29	0.37	0.36	0.12	0.05	0.22	0.08	0.03	0.07	0.01	0.05
Qwen2.5-7B-128k	0.18	0.32	0.43	0.41	0.34	0.40	0.38	0.14	0.05	0.24	0.08	0.03	0.07	0.01	0.04
Qwen2.5-14B-128k	0.19	0.33	0.43	0.43	0.35	0.42	0.41	0.15	0.06	0.26	0.08	0.03	0.07	0.01	0.05
Qwen2.5-32B-128k	0.21	0.36	0.43	0.46	0.36	0.43	0.40	0.16	0.06	0.26	0.11	0.03	0.09	0.01	0.06
Llama-7B-2k	0.03	0.07	0.17	0.08	0.09	0.06	0.05	0.02	0.02	0.04	0.03	0.01	0.04	0.00	0.02
Llama2-7B-4k	0.05	0.10	0.20	0.11	0.11	0.06	0.06	0.02	0.02	0.04	0.03	0.01	0.04	0.00	0.02
Llama3-8B-8k	0.03	0.10	0.17	0.13	0.11	0.03	0.05	0.01	0.03	0.03	0.09	0.02	0.04	0.00	0.04
Llama-30B-2k	0.04	0.09	0.18	0.11	0.10	0.05	0.05	0.02	0.02	0.04	0.05	0.01	0.04	0.00	0.02
OPT-30B-2k	0.02	0.07	0.16	0.08	0.08	0.04	0.05	0.01	0.02	0.03	0.03	0.00	0.02	0.00	0.01
Wizard-Vicuna-2k	0.10	0.18	0.33	0.17	0.20	0.24	0.20	0.06	0.02	0.13	0.04	0.01	0.04	0.00	0.02
LwQ-Instruct-2k	0.16	0.22	0.34	0.19	0.23	0.34	0.24	0.11	0.02	0.18	0.06	0.02	0.08	0.00	0.04
30B-Epsilon-2k	0.13	0.19	0.30	0.16	0.20	0.24	0.17	0.08	0.02	0.13	0.05	0.02	0.07	0.01	0.04

Table 4: Results on single-doc QA, multi-doc QA and summarization tasks. The indexes, like "1-1" or "4-1", are following Table 1. "avg" represents the average performance of subtasks under different main tasks.
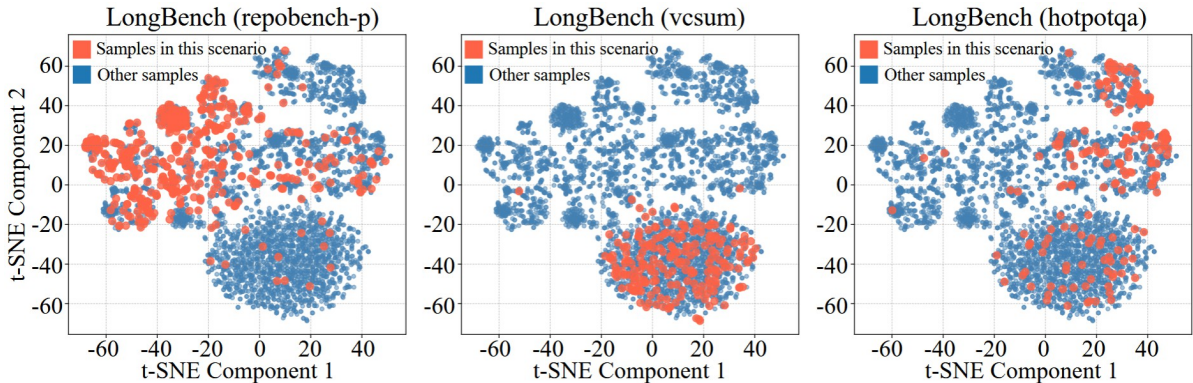
Figure 11: The visualization of learned representation (e_j, β_j) of test sample.

enhancing the efficiency and memory retention of Transformers (Tay et al., 2022). This includes innovations in sparse and efficient computation (Child et al., 2019; Kitaev et al., 2020; Beltagy et al., 2020; Zaheer et al., 2020; Wang et al., 2020; Fedus et al., 2022; Ding et al., 2023), as well as the integration of recurrent and memory modules (Dai et al., 2019; Rae et al., 2020; Wu et al., 2022; Martins et al., 2022; Bulatov et al., 2022; Orvieto et al., 2023; Liang et al., 2023; Zhou et al., 2023).

More recently, several advanced methods (Press et al., 2022; Sun et al., 2022; Chen et al., 2023) have been developed to facilitate length extrapolation in Transformers. These techniques have been incorporated into the training frameworks of long-context LLMs such as ChatGLM2-32k (Zeng et al., 2023) and LongChat-32k (Li et al., 2023a), among others. These models have successfully extended their context lengths to 128k tokens or more (An-

thropic, 2024; OpenAI, 2024; Reid et al., 2024; GLM et al., 2024; Dubey et al., 2024; Xiong et al., 2024; Fu et al., 2024; Bai et al., 2024b; Gao et al., 2024), marking a significant advancement in the field.

D.2 The LCU Benchmarks for LLMs

Given the critical importance of LCU capabilities for LLMs, an increasing number of benchmarks have been proposed to evaluate these capabilities, playing a pivotal role in exploring and advancing LLMs' LCU proficiency. A significant portion of these benchmarks of LLMs focuses on comprehensive LCU assessment, encompassing tasks such as Question Answering, information retrieval, and summarization. Notable examples include L-Eval (An et al., 2024), LongBench (Bai et al., 2024c), ZeroSCROLLS (Shaham et al., 2023), BAMBOO (Dong et al., 2024), LooGLE (Li

Model	Few-shot Learning					Synthetic				Code			Overall		
	4-1	4-2	4-3	4-4	Avg	5-1	5-2	5-3	Avg	6-1	6-2	Avg	EN	ZH	All
DeepSeek-V3-128k	0.77	0.93	0.63	0.35	0.67	0.05	0.38	0.34	0.26	0.90	0.86	0.88	0.43	0.26	0.45
GPT-4o-mini-128k	0.76	0.93	0.60	0.32	0.65	0.04	0.35	0.28	0.23	0.90	0.85	0.87	0.42	0.24	0.43
GPT-3.5-Turbo-16k	0.70	0.88	0.49	0.23	0.58	0.03	0.27	0.21	0.17	0.84	0.78	0.81	0.37	0.19	0.37
Internlm3-8B-32k	0.61	0.86	0.48	0.17	0.53	0.01	0.18	0.16	0.12	0.67	0.62	0.64	0.32	0.16	0.32
ChatGLM3-6B-8k	0.47	0.69	0.26	0.13	0.39	0.00	0.03	0.04	0.02	0.49	0.48	0.48	0.19	0.09	0.19
ChatGLM4-9B-128k	0.69	0.89	0.51	0.24	0.58	0.02	0.24	0.16	0.14	0.79	0.70	0.75	0.36	0.18	0.36
Qwen-7B-8k	0.58	0.78	0.34	0.20	0.48	0.01	0.12	0.08	0.07	0.74	0.68	0.71	0.25	0.11	0.27
Qwen2-7B-128k	0.66	0.89	0.44	0.20	0.55	0.01	0.15	0.11	0.09	0.74	0.69	0.71	0.32	0.14	0.32
Qwen2.5-7B-128k	0.65	0.87	0.49	0.21	0.55	0.01	0.19	0.16	0.12	0.69	0.65	0.67	0.33	0.17	0.33
Qwen2.5-14B-128k	0.65	0.87	0.49	0.22	0.56	0.01	0.19	0.16	0.12	0.68	0.64	0.66	0.33	0.17	0.33
Qwen2.5-32B-128k	0.69	0.90	0.54	0.25	0.59	0.02	0.24	0.19	0.15	0.76	0.71	0.74	0.36	0.20	0.36
Llama-7B-2k	0.47	0.71	0.21	0.15	0.39	0.00	0.04	0.03	0.02	0.64	0.58	0.61	0.18	0.06	0.20
Llama2-7B-4k	0.53	0.75	0.22	0.15	0.41	0.00	0.07	0.05	0.04	0.69	0.63	0.66	0.20	0.07	0.21
Llama3-8B-8k	0.60	0.74	0.30	0.24	0.47	0.01	0.10	0.14	0.08	0.74	0.68	0.71	0.22	0.11	0.24
Llama-30B-2k	0.50	0.72	0.24	0.17	0.41	0.00	0.08	0.05	0.04	0.67	0.62	0.65	0.20	0.07	0.21
OPT-30B-2k	0.40	0.65	0.16	0.11	0.33	0.00	0.02	0.03	0.02	0.49	0.47	0.48	0.15	0.05	0.16
Wizard-Vicuna-2k	0.52	0.76	0.24	0.13	0.41	0.01	0.08	0.05	0.05	0.62	0.55	0.59	0.23	0.08	0.23
LwQ-Instruct-2k	0.54	0.78	0.30	0.16	0.45	0.01	0.13	0.07	0.07	0.73	0.66	0.70	0.28	0.09	0.28
30B-Epsilon-2k	0.58	0.83	0.34	0.18	0.48	0.01	0.13	0.07	0.07	0.76	0.70	0.73	0.27	0.09	0.27

Table 5: Results on few-shot learning, synthetic, and code tasks. 'Overall' is computed by the macro-average (the mean of 'Avg') over major task categories. This is computed on English (EN) tasks, Chinese (ZH) tasks, and all (All) tasks, code tasks are included in both languages. The indexes, like "1-1" or "4-1", are following Table 1. "avg" represents the average performance of subtasks under different main tasks.

et al., 2023b), ∞-bench (Zhang et al., 2024b), Ruler (Hsieh et al., 2024), and HELMET (Yen et al., 2024). Another category of benchmarks is specifically designed to explore particular aspects of LCU capabilities. These include retrieval and attribution tasks (Kamradt, 2023; Kuratov et al., 2024; Song et al., 2024; Laban et al., 2024; Zhang et al., 2024a; Vodrahalli et al., 2024; Krishna et al., 2024), document QA (Dua et al., 2019; Dasigi et al., 2021; Pang et al., 2022; Wang et al., 2024), summarization (Zhong et al., 2021; Huang et al., 2021a; Wang et al., 2022), coding (Liu et al., 2023; Bogomolov et al., 2024), many-shot learning (Agarwal et al., 2024), and long-text generation (Bai et al., 2024e; Wu et al., 2024; Liu et al., 2024; Que et al., 2024).

These specialized benchmarks provide targeted insights into the diverse and complex facets of LCU, contributing to a more nuanced understanding and development of LLMs' long-context processing abilities.

D.3 Low-cost Deep Learning

Recently, there has been a surge of efforts aimed at achieving low-cost deep learning, encompassing strategies such as the compression of model parameters or the design of lightweight architectures (Yang et al., 2024; Muralidharan et al., 2024; Lin et al., 2024; Kim et al., 2024; Zhong et al., 2023b; He et al., 2021; Huang et al., 2022,

2021b; Liang et al., 2020). Concurrently, some research has explored compressing the training dataset (Gadre et al., 2024; Sachdeva and McAuley, 2023; Yu et al., 2023; Lei and Tao, 2023; Touvron et al., 2021) to reduce computational costs while maintaining performance. Beyond these approaches, in the era of large language models, works including this paper consider compressing test data (Polo et al., 2024; Pacchiardi et al., 2024; Kipnis et al., 2024) as an effective means to aid in model architecture design, parameter tuning, and other training-related processes, thereby accelerating the iteration speed of robust models.

E The Visualization of Learned Representation of Test Samples

In Section 3, we use the performance record of m LLMs on various test samples in LongBench, and use a logistic regression model for representation learning, obtaining their representations (e_j, β_j). In Fig. 11, we visualize test samples from certain sub-tasks listed in Table 1 using t-SNE (Van der Maaten and Hinton, 2008). It can be observed that many test samples form clusters, and the representations of samples within the same cluster are highly similar. This further demonstrates that Long-Bench contains a significant amount of redundancy in its data, and the representation learning method proposed in Section 3 is effective for identifying
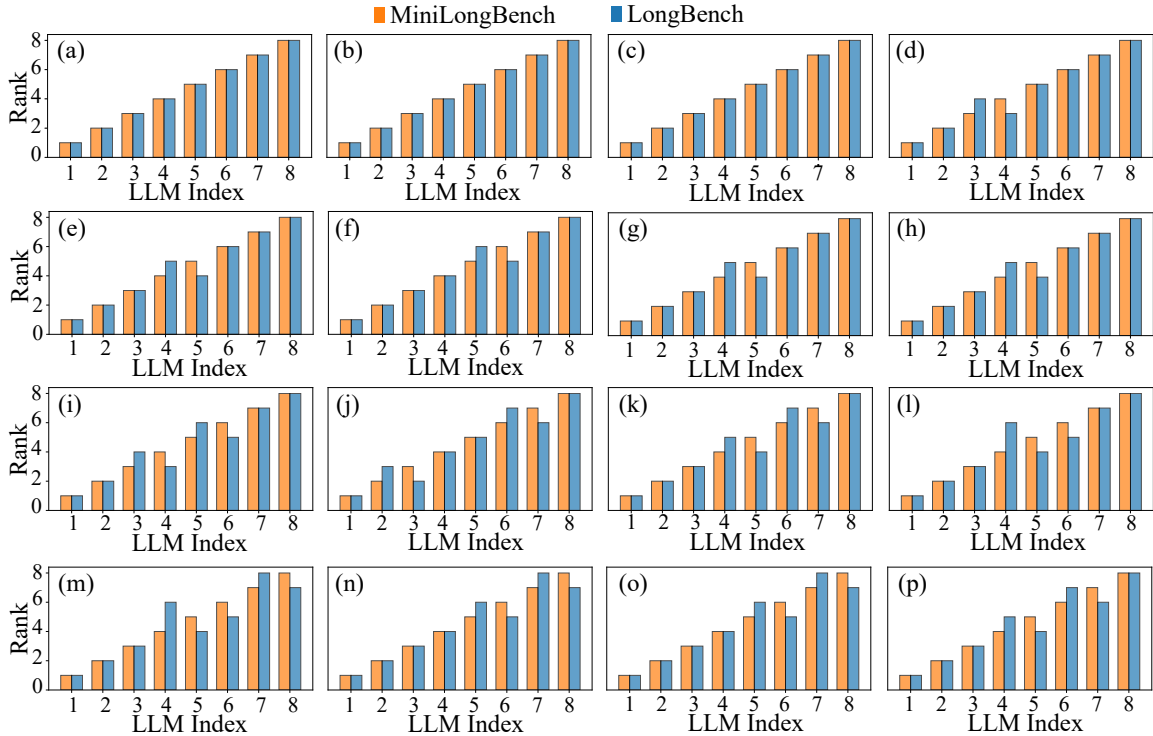
Figure 12: The more examples of visualization of ranking by MiniLongBench and LongBench..

F More Visualizations of Ranking

redundant data in LongBench through clustering.

In the Fig. 10, we provided some examples of rankings by MiniLongBench and LongBench. In this section, we will present more random examples to illustrate the usability and reliability of Mini-LongBench. The results are shown in Fig. 12. As illustrated in Fig. 12, similar to the observations in the main text's Fig. 10, the results from 16 random sampling trials consistently demonstrate that the ranking outcomes of various LLMs on Mini-LongBench closely align with those on LongBench. Although minor discrepancies exist, they are within an acceptable range, particularly considering that the Spearman correlation coefficient (Sp) does not reach a perfect 1.0. These visualizations further validate that MiniLongBench achieves evaluation results comparable to LongBench while significantly reducing computational costs. This highlights MiniLongBench's effectiveness as a low-cost alternative for assessing LLM performance.

G Evaluating Directly by MiniLongBench

In Section 4.2 of the main text, we primarily introduce a method that utilizes test samples from Mini-LongBench to assist in evaluating the performance of a target LLM on LongBench. This method

Model	SQA	MQA	SUM	FSL	SYN	CODE
DeepSeek-V3-128k	0.58	0.61	0.17	0.72	0.67	0.72
GPT-4o-mini-128k	0.54	0.57	0.18	0.61	0.67	0.70
GPT-3.5-Turbo-16k	0.43	0.47	0.17	0.40	0.64	0.61
Internlm3-8B-32k	0.38	0.55	0.16	0.34	0.67	0.19
ChatGLM3-6B-8k	0.31	0.15	0.17	0.30	0.07	0.36
ChatGLM4-9B-128k	0.48	0.56	0.18	0.56	0.58	0.36
Qwen-7B-8k	0.30	0.22	0.14	0.34	0.20	0.65
Qwen2-7B-128k	0.42	0.42	0.16	0.49	0.36	0.43
Qwen2.5-7B-128k	0.48	0.50	0.17	0.60	0.67	0.10
Qwen2.5-14B-128k	0.46	0.56	0.18	0.65	0.67	0.01
Qwen2.5-32B-128k	0.50	0.53	0.17	0.71	0.67	0.18
Llama-7B-2k	0.13	0.09	0.12	0.25	0.05	0.57
Llama2-7B-4k	0.16	0.10	0.10	0.23	0.14	0.59
Llama3-8B-8k	0.15	0.08	0.10	0.35	0.21	0.70
Llama-30B-2k	0.15	0.08	0.09	0.25	0.15	0.64
OPT-30B-2k	0.11	0.09	0.11	0.15	0.07	0.44
Wizard-Vicuna-2k	0.29	0.26	0.16	0.27	0.16	0.43
LwQ-Instruct-2k	0.34	0.36	0.16	0.31	0.18	0.61
30B-Epsilon-2k	0.26	0.21	0.17	0.42	0.20	0.62

Table 6: Specific evaluation results on evaluating directly by MiniLongBench. Due to differences in evaluation methods, the score values presented in this table vary somewhat from those in Table 2, but they yield a similar ranking of LLMs in terms of LCU capability.

achieves a performance of up to 0.97 in Sp. In practice, we can also directly test the target LLM on MiniLongBench's test samples to obtain an assessment of its LCU capability. The results in Fig. 13 confirm that this evaluation method achieves good Sp across various tasks in MiniLongBench, with an average Sp of 0.95, slightly lower than the evaluation method presented in Section 4.2. Furthermore, in Table 6, we present the results of this evaluation
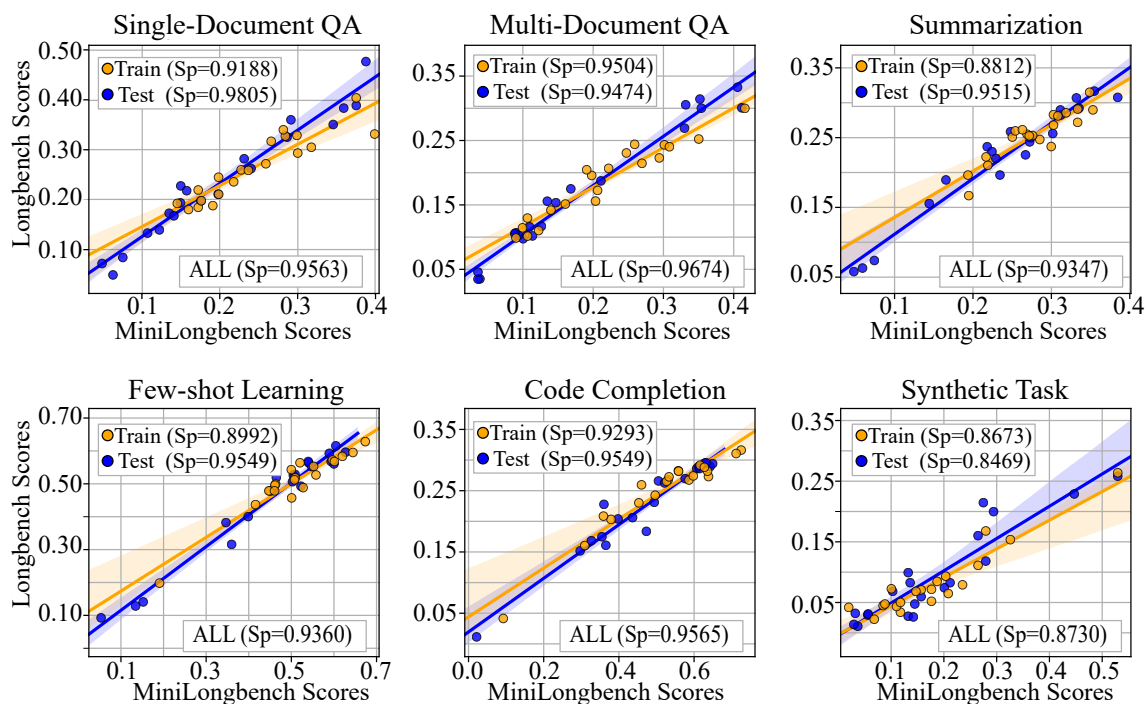
Figure 13: The analysis of rank correlation (Sp) between LongBench and MiniLongBench where the result of MiniLongBench is evaluating directly.

method across six main tasks. Additionally, we provide more detailed results for each subtask in Table 7 and Table 8.

It is noteworthy that, in practice, whether directly evaluating on LongBench or MiniLongBench, or using the predictive method in Section 4.2, there may be some discrepancies in the score values. However, these discrepancies do not affect the ranking of LLMs' LCU capabilities. For instance, Fig. 13 and the main text's Fig. 5 demonstrate that the results from different evaluation methods are highly consistent, despite minor deviations in score values. This phenomenon primarily arises from several factors: first, MiniLongBench involves significant pruning of test samples compared to Long-Bench, leading to unavoidable errors; second, during the logistic regression in Section 3's Eq. (2), normalization and discretization introduce certain errors, particularly in scaling. Fortunately, the primary goal of the LCU benchmark is to rank LLMs based on their LCU capabilities, so the absolute score values do not impact the final outcomes.

H   The Cost by Fine-tuning $\bar{\theta}$

In Section 4.2, additional fine-tuning of $\bar{\theta}$ is required, which primarily involves two costs: the training cost for fine-tuning and the storage cost for the representation vectors of LongBench's test samples. In practice, these costs are minimal and

entirely acceptable. Specifically, storing the test samples of MiniLongBench and the representation vectors of LongBench's test samples requires only 9.01MB and 1.13MB of disk space, respectively. This is significantly lower and entirely acceptable compared to the original storage cost of nearly 200MB for LongBench's test samples. This reduction is largely due to the two-step dimensionality compression method described in Section 3, which uses text embedding and PCA to compress each feature vector to a dimension of just $d = 10$, thereby greatly reducing storage costs.

On the other hand, the cost of fine-tuning $\bar{\theta}$ is also very low and can even be performed on a standard laptop without the need for server-grade GPUs. This is because MiniLongBench contains only about 200 test samples, and the dimensions of all representation vectors are all $d = 10$, so the logistic regression training does not require significant computational power. Through 100 repeated experiments, the average time required for fine-tuning $\bar{\theta}$ was calculated. on a server (CPU: AMD EPYC 7K62, GPU: RTX 3090 24GB) and a laptop (CPU: AMD Ryzen 6 5600H, GPU: RTX 3050 4GB), fine-tuning takes approximately 0.02 seconds and 0.03 seconds, respectively. Compared to the original testing time of LongBench shown in Fig. 1, this is almost negligible.

Model	Single-Doc QA					Multi-Doc QA					Summarization				
	1-1	1-2	1-3	1-4	Avg	2-1	2-2	2-3	2-4	Avg	3-1	3-2	3-3	3-4	Avg
DeepSeek-V3-128k	0.66	0.40	0.53	0.72	0.58	0.74	0.89	0.59	0.22	0.61	0.18	0.21	0.16	0.15	0.17
GPT-4o-mini-128k	0.51	0.47	0.52	0.67	0.54	0.64	0.81	0.54	0.29	0.57	0.16	0.20	0.17	0.18	0.18
GPT-3.5-Turbo-16k	0.12	0.53	0.48	0.60	0.43	0.51	0.72	0.44	0.20	0.47	0.14	0.20	0.17	0.17	0.17
Internlm3-8B-32k	0.07	0.36	0.44	0.64	0.38	0.59	0.77	0.63	0.22	0.55	0.18	0.17	0.16	0.14	0.16
ChatGLM3-6B-8k	0.17	0.28	0.38	0.42	0.31	0.11	0.20	0.12	0.15	0.15	0.15	0.18	0.16	0.18	0.17
ChatGLM4-9B-128k	0.33	0.40	0.56	0.62	0.48	0.63	0.76	0.67	0.20	0.56	0.16	0.21	0.16	0.19	0.18
Qwen-7B-8k	0.16	0.28	0.46	0.32	0.30	0.29	0.22	0.22	0.13	0.22	0.17	0.16	0.17	0.06	0.14
Qwen2-7B-128k	0.38	0.40	0.38	0.52	0.42	0.64	0.71	0.19	0.15	0.42	0.18	0.18	0.15	0.13	0.16
Qwen2.5-7B-128k	0.50	0.37	0.46	0.58	0.48	0.71	0.66	0.50	0.14	0.50	0.18	0.21	0.14	0.15	0.17
Qwen2.5-14B-128k	0.41	0.43	0.44	0.56	0.46	0.67	0.89	0.44	0.23	0.56	0.16	0.23	0.15	0.16	0.18
Qwen2.5-32B-128k	0.45	0.43	0.42	0.68	0.50	0.66	0.76	0.48	0.21	0.53	0.16	0.21	0.15	0.17	0.17
Llama-7B-2k	0.03	0.11	0.23	0.12	0.13	0.07	0.08	0.15	0.04	0.09	0.14	0.05	0.18	0.12	0.12
Llama2-7B-4k	0.13	0.20	0.23	0.09	0.16	0.05	0.11	0.13	0.10	0.10	0.16	0.07	0.04	0.14	0.10
Llama3-8B-8k	0.02	0.20	0.17	0.22	0.15	0.04	0.10	0.13	0.06	0.08	0.18	0.14	0.01	0.09	0.10
Llama-30B-2k	0.03	0.17	0.27	0.11	0.15	0.07	0.12	0.08	0.04	0.08	0.16	0.05	0.05	0.11	0.09
OPT-30B-2k	0.03	0.18	0.11	0.11	0.11	0.06	0.09	0.14	0.07	0.09	0.18	0.05	0.14	0.08	0.11
Wizard-Vicuna-2k	0.28	0.21	0.49	0.16	0.29	0.46	0.26	0.19	0.13	0.26	0.16	0.16	0.18	0.14	0.16
LwQ-Instruct-2k	0.33	0.36	0.46	0.20	0.34	0.58	0.40	0.37	0.09	0.36	0.17	0.18	0.19	0.10	0.16
30B-Epsilon-2k	0.24	0.21	0.40	0.18	0.26	0.27	0.22	0.28	0.07	0.21	0.14	0.18	0.18	0.17	0.17

Table 7: Results on single-doc QA, multi-doc QA and summarization tasks based on evaluating directly by MiniLongBench. The indexes, like "1-1" or "4-1", are following Table 1. "avg" represents the average performance of subtasks under different main tasks.

Model	Few-show Learning					Synthetic				Code			Overall		
	4-1	4-2	4-3	4-4	Avg	5-1	5-2	5-3	Avg	6-1	6-2	Avg	EN	ZH	All
DeepSeek-V3-128k	0.63	1.00	0.38	0.88	0.72	0.00	1.00	1.00	0.67	0.83	0.62	0.72	0.52	0.59	0.58
GPT-4o-mini-128k	0.63	1.00	0.31	0.50	0.61	0.00	1.00	1.00	0.67	0.80	0.61	0.70	0.49	0.53	0.54
GPT-3.5-Turbo-16k	0.63	0.60	0.38	0.00	0.40	0.00	0.93	1.00	0.64	0.73	0.49	0.61	0.42	0.39	0.45
Internlm3-8B-32k	0.00	1.00	0.38	0.00	0.34	0.00	1.00	1.00	0.67	0.19	0.18	0.19	0.36	0.40	0.38
ChatGLM3-6B-8k	0.50	0.01	0.30	0.38	0.30	0.00	0.00	0.20	0.07	0.45	0.27	0.36	0.19	0.26	0.22
ChatGLM4-9B-128k	0.63	0.83	0.35	0.44	0.56	0.00	1.00	0.73	0.58	0.59	0.13	0.36	0.44	0.44	0.45
Qwen-7B-8k	0.44	0.25	0.16	0.50	0.34	0.00	0.40	0.20	0.20	0.76	0.55	0.65	0.28	0.24	0.31
Qwen2-7B-128k	0.38	0.96	0.35	0.27	0.49	0.00	0.47	0.60	0.36	0.54	0.32	0.43	0.37	0.34	0.38
Qwen2.5-7B-128k	0.50	0.83	0.37	0.69	0.60	0.00	1.00	1.00	0.67	0.10	0.11	0.10	0.39	0.51	0.42
Qwen2.5-14B-128k	0.63	0.86	0.35	0.75	0.65	0.00	1.00	1.00	0.67	0.00	0.03	0.01	0.39	0.54	0.42
Qwen2.5-32B-128k	0.63	0.97	0.35	0.88	0.71	0.00	1.00	1.00	0.67	0.29	0.07	0.18	0.41	0.59	0.46
Llama-7B-2k	0.31	0.31	0.12	0.25	0.25	0.00	0.07	0.07	0.05	0.71	0.44	0.57	0.18	0.12	0.20
Llama2-7B-4k	0.25	0.25	0.14	0.29	0.23	0.00	0.20	0.22	0.14	0.71	0.48	0.59	0.18	0.17	0.22
Llama3-8B-8k	0.38	0.20	0.16	0.69	0.35	0.00	0.17	0.47	0.21	0.84	0.57	0.70	0.19	0.30	0.27
Llama-30B-2k	0.31	0.24	0.15	0.31	0.25	0.00	0.32	0.13	0.15	0.76	0.52	0.64	0.19	0.14	0.23
OPT-30B-2k	0.17	0.21	0.11	0.13	0.15	0.02	0.10	0.09	0.07	0.51	0.37	0.44	0.14	0.09	0.16
Wizard-Vicuna-2k	0.25	0.42	0.17	0.23	0.27	0.00	0.27	0.20	0.16	0.51	0.35	0.43	0.26	0.17	0.26
LwQ-Instruct-2k	0.31	0.31	0.17	0.44	0.31	0.00	0.33	0.20	0.18	0.71	0.52	0.61	0.32	0.21	0.33
30B-Epsilon-2k	0.24	0.80	0.26	0.38	0.42	0.00	0.40	0.20	0.20	0.75	0.50	0.62	0.30	0.20	0.31

Table 8: Results on few-shot learning, synthetic, and code tasks based on evaluating directly by MiniLongBench. 'Overall' is computed by the macro-average (the mean of 'Avg') over major task categories. This is computed on English (EN) tasks, Chinese (ZH) tasks, and all (All) tasks, code tasks are included in both languages. The indexes, like "1-1" or "4-1", are following Table 1. "avg" represents the average performance of subtasks under different main tasks.