Long Context is Not Long at All: A Prospector of Long-Dependency Data for Large Language Models

Longze Chen[1,2]* **Ziqiang Liu**[1,2]* **Wanwei He**[1,2]* **Yinhe Zheng**
Yunshui Li[1,2] **Run Luo**[1,2] **Min Yang**[1]†
[1]Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
{lz.chen2, zq.liu4, ww.he, min.yang}@siat.ac.cn

Abstract

Long-context modeling capabilities are important for large language models (LLMs) in various applications. However, directly training LLMs with long context windows is insufficient to enhance this capability since some training samples do not exhibit strong semantic dependencies across long contexts. In this study, we propose a data mining framework **ProLong**[1] that can assign each training sample with a long dependency score, which can be used to rank and filter samples that are more advantageous for enhancing long-context modeling abilities in LLM training. Specifically, we first use delta perplexity scores to measure the *Dependency Strength* between text segments in a given document. Then, we refine this metric based on the *Dependency Distance* of these segments to incorporate spatial relationships across long contexts. Final results are calibrated with a *Dependency Specificity* metric to prevent trivial dependencies introduced by repetitive patterns. Moreover, a random sampling approach is proposed to optimize the computational efficiency of ProLong. Comprehensive experiments on multiple benchmarks indicate that ProLong effectively identifies documents that carry long dependencies, and LLMs trained on these documents exhibit significantly enhanced long-context modeling capabilities.

1 Introduction

Large language models (LLMs) are widely used in many natural language processing (NLP) tasks (Brown et al., 2020). These tasks often require dealing with long text inputs (Bai et al., 2023; Zhang et al., 2023), such as lengthy documents (Zhou et al., 2022), long conversation histories in chatbots (Zhong et al., 2023) or large codebases (Guo et al., 2023). Therefore, enhancing LLMs to model long-context inputs is a prominent desiderata.

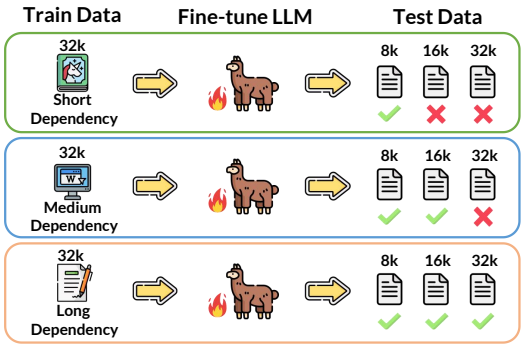

Figure 1: Samples that carry longer dependencies better enhance LLMs' long-context modeling capabilities, even with a fixed training context window of 32k.

There are primarily two categories of approaches to expand the context window of an LLM. The first category fine-tunes LLMs with longer context windows (Chen et al., 2023a), while the second category adjusts the LLM's positional encoding or attention mechanism to accommodate larger position indices without further re-training (Press et al., 2021; bloc97, 2023). However, non-training methods often produce results inferior to those of fine-tuned LLMs (Xiong et al., 2023), which model long-context inputs more effectively and generally achieve lower perplexity scores.

Although reported to be feasible, simply fine-tuning LLMs with naively sampled long corpora does not ensure improved long context modeling capabilities (de Vries, 2023). Some of these fine-tuned LLMs may still struggle to effectively process and utilize information from long input contexts even if they obtain a decently low perplexity score (Sun et al., 2021; Pal et al., 2023). This can lead to low performance in various downstream applications, even in some basic synthetic retrieval tasks (Liu et al., 2023). Nevertheless, few approaches try to tackle this long-context modeling

*Equal contribution.
†Corresponding author.
[1] https://github.com/October2001/ProLong

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 8222–8234

issue from a data-centric perspective.

As revealed by Fu et al. (2023), the quality of fine-tuning data plays a critical role in enhancing the long-context modeling capabilities of LLMs (Li et al., 2023b). Besides, de Vries (2023) also report that high-quality corpora significantly outperform other factors in boosting long-context performance. Upon further exploration, we recognize that high-quality long-text data is characterized by the presence of **long-range dependencies**. The importance of encapsulating long-range dependencies in LLMs is also underscored by Borgeaud et al. (2022), which elucidates the benefits of integrating global dependencies into retrieval-augmented language models.

However, such strong semantic dependencies are rare in typical training samples (de Vries, 2023) and diminish as the distance between segments increases (Staniszewski et al., 2023). Even with identical sequence lengths, different samples may exhibit varying dependency density. Specifically, certain long training samples often comprise concatenated short documents that are randomly selected and do not have any semantic dependencies. Moreover, even for inherently long documents, like novels, most tokens depend only to a brief span of preceding context. This phenomenon can be simply concluded as "**long context is not long at all**", leading to challenges in model learning (Figure 1). Therefore, we argue that explicitly incorporating long-dependency data into the fine-tuning process can facilitate long-context modeling.

In this paper, we propose a novel framework (called ProLong) to mine long-dependency data. ProLong assigns a long-dependency score to each document, which serves as an indicator of the dependency density across long contexts. Documents with higher scores are deemed more advantageous for boosting long context modeling, therefore we can use these scores to rank and filter high quality corpus for LLM fine-tuning. Concretely, ProLong first partitions each document into fixed-length segments and evaluates dependency relationships between each segment pair from three perspectives: (i) *dependency strength* quantifies the difference in perplexities of a given segment when conditioned with or without its preceding segments. This metric measures the contribution of the preceding segment to the current one; (ii) *dependency distance* measures the positional gap and spatial relationship between two text segments; and (iii) *dependency specificity* employs entropy to ensure a non-uniform distribution of dependency strengths across all preceding segments, mitigating trivial dependencies introduced by repetitive patterns. A dependency score is assigned to each segment pair by combining the above three perspectives, and a final long-dependency score for the entire document is computed by accumulating dependency scores of all segment pairs.

Further, we adopt various strategies to optimize the computational efficiency of ProLong, including sampling among segments, evaluating perplexity scores with small models and curating test sets for rapid validation. Experiments on multiple benchmarks indicate that ProLong effectively identifies documents that carry long dependencies and LLMs trained on these documents exhibit significantly enhanced long-context modeling capabilities. Our contributions are summarized as follows:

1. To the best of our knowledge, this is the first study to explore the relationship between dependency density and the quality of long-text data.

2. We propose ProLong, a data mining framework for identifying long-dependency data. With ProLong, significant performance boosts are observed using only 50% of fine-tuning data.

3. We provide an in-depth analysis of ProLong's components, optimizing computational efficiency and making it practical for large-scale corpora.

4. We develop two models, ProLong-7b/13b, using training samples derived from the ProLong framework. Experiments show that our models outperform equal-sized competitors on both language modeling and real long-context tasks.

2 Related Work

Long-context LLMs. Large Language Models (LLMs), such as Llama (Touvron et al., 2023a) with a context size of 2048 and Llama2 (Touvron et al., 2023b) with 4096, are typically pre-trained using a pre-determined context size. However, training LLMs from scratch with extended contexts is extremely time-consuming and labor-intensive. Therefore, recent research endeavors to extend the limited context length of these models via fine-tuning. Fine-tuning methods such as Positional Interpolation (Chen et al., 2023a), "NTK-aware" interpolation (bloc97, 2023), Giraffe (Pal et al., 2023) and YaRN (Peng et al., 2023) modify rotary position encoding (RoPE) (Su et al., 2024) and then fine-tune the model with a small number of training steps to expand the context window. De-

spite this, full fine-tuning is still computationally expensive. Consequently, researchers have pivoted towards exploring methods to reduce training costs. For instance, LongLora (Chen et al., 2023b) proposes S^2-Attn and utilizes LoRA (Hu et al., 2021) for low-cost and efficient training; Soaring (Zhang et al., 2024) introduces Activation Beacon, achieving a dramatic extension of LLMs context at low training costs. Nonetheless, these methods have not taken into account the quality of the long-context data used during fine-tuning stage. In contrast, our approach emphasizes the high-quality long-context training data, facilitating the efficient extension of LLMs' context with a limited quantity of data.

Constructing Long-context Training Data. Long-context data is not simply the arbitrary concatenation of unrelated short texts. The construction of high-quality or specialized forms of long-context data also holds significant research value. de Vries (2023) argues against squandering attention on randomly connected long texts and instead proposes considering strategies for obtaining meaningful long-context data. SPLICE (Staniszewski et al., 2023) introduces structured packing for long-context, constructing training examples by assembling multiple similar documents selected through a retrieval method. Ziya-Reader (Junqing et al., 2023) constructs a tailored Multi-doc QA task that requires concentration on different positions in contexts to address the "lost in the middle" (Liu et al., 2023) problem. Yu (2023) delves into the required data format for training long-context models, claiming that the use of "W-shaped data" is essential to tackle the "lost in the middle" problem. Upon further exploration, we identify that high-quality long-text data is characterized by the presence of long-range dependencies. Our method focuses on selecting these long-dependency data from a substantial corpus of long-context data.

3 ProLong Framework

Figure 2 presents the overall framework of ProLong, which assigns a Long Dependency Score (LDS) for a given data sample S (S could be an intact document or a concatenated training sample). Specifically, we first divide S into N segments of equal length $c_1, ..., c_N$. Then for each segment pair $(c_j, c_i), j < i$, three scores are calculated: 1. The Dependency Strength (DST) score evaluates how much semantic dependencies are there between c_i and c_j; 2. The Dependency Distance (DDI) score demonstrates the distance between c_i and c_j; 3. The Dependency Specificity (DSP) score penalizes the erroneous gains introduced by similar patterns in each segments. The final LDS for S is obtained by merging the above three sets of scores.

3.1 Dependency Strength

For a given segment pair $(c_j, c_i), j < i$, the dependency between c_i and c_j can be evaluated by the delta perplexity score of c_i when c_j is given or not in the context. Specifically, if there are strong semantic dependencies between c_i and c_j, then it is expected that the perplexity score of c_i will be dramatically reduced if c_j is presented in the context. Therefore, we can select a language model and use the difference of perplexity scores to quantify the dependency strength $\mathrm{DST}_{i,j}$ between c_i and c_j:

$$\mathrm{DST}_{i,j} = \frac{\mathrm{PPL}(c_i) - \mathrm{PPL}(c_i \mid c_j)}{\mathrm{PPL}(c_i)} \quad (1)$$

where $\mathrm{PPL}(c_i)$ is the perplexity score of c_i without any context, and $\mathrm{PPL}(c_i \mid c_j)$ is the conditional perplexity score of c_i when c_j is provided in the input context. In this study, we use a small-sized and fixed model to calculate perplexity scores.

3.2 Dependency Distance

Another factor to consider in long dependency relationships is the distance between c_i and c_j. Specifically, distant segments are more important for learning long-range dependencies. Therefore for each pair $(c_j, c_i), j < i$, we consider a dependency distance score $\mathrm{DDI}_{i,j}$ to measure the positional gap:

$$\mathrm{DDI}_{i,j} = \frac{i - j}{N - 1} \quad (2)$$

where N is the total segment count.

3.3 Dependency Specificity

Our early experiments observe that some documents in the training corpus contain repeated text spans. Specifically, in the extreme case, the entire document contains only one unique token. For a given segment pair $(c_j, c_i), j < i$ from these documents, c_i and c_j are nearly the same and c_i will receive nearly perfect conditional perplexity score if c_j is presented in the context. In this case, we will achieve an extremely high dependency strength score $\mathrm{DST}_{i,j}$ between c_i and c_j.

However, such trivial dependencies brought by repeated text spans are usually harmful to the training of LLMs (Lee et al., 2021). Therefore, we introduce the concept of dependency specificity (DSP)

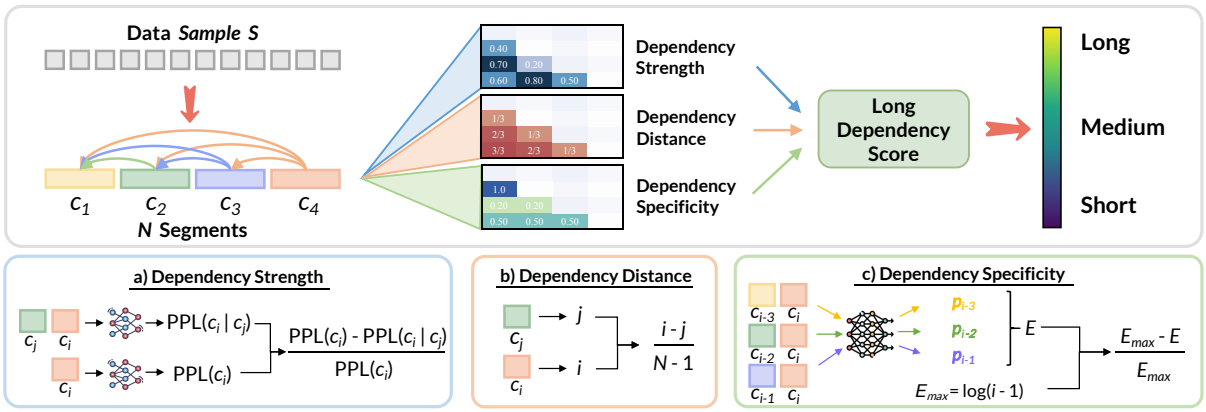

Figure 2: ProLong first segments a training sample S into N equal-length portions, then computing three key metrics: (a) dependency strength, (b) dependency distance, and (c) dependency specificity. These metrics are integrated via Eq.6 to derive the long dependency score for S.

to mitigate this issue. Specifically, for a given segment c_i, we can calculate the reduction of perplexity scores $\Delta\text{PPL}_{i,j} = \text{PPL}(c_i) - \text{PPL}(c_i|c_j)$ when conditioning c_i on each of its preceding segments c_j, $(j = 1, ..., i-1)$. A probability distribution $P(c_i) = (p_1, ..., p_{i-1})$ can then be obtained by applying Softmax on $\Delta\text{PPL}_{i,j}$:

$$p_j = \frac{\exp\left(\Delta\text{PPL}_{i,j}\right)}{\sum_{k=1}^{i-1}\exp\left(\Delta\text{PPL}_{i,k}\right)} \tag{3}$$

$P(c_i)$ denotes the dependency distribution of c_i on all preceding segments. A uniform $P(c_i)$ indicates that c_i relies almost equally on each preceding segment c_j $(j = 1, ..., i-1)$, suggesting that c_j are nearly identical. Consequently, the entropy of $P(c_i)$ can serve to adjust the DST score derived from Eq.1, addressing the issue of repeated spans.

In this study, we define the dependency specificity DSP_i score for a given segment c_i as

$$\text{DSP}_i = \frac{E_{max} - E}{E_{max}} \tag{4}$$

where $E_{max} = \log(i-1)$ is the entropy of a $i-1$ dimensional uniform distribution. And E is the entropy of $P(c_i)$, i.e., $E = -\sum_{j=1}^{i-1} p_j \log(p_j)$. A higher DSP_i value suggests that the segment c_i does not depend equally on all its preceding segments, indicating a lower likelihood of these segments being repeated spans.

3.4 Long Dependency Scores

We define a long dependency score between each segment pair $(c_j, c_i), j < i$ as:

$$\text{LDS}_{i,j} = (\alpha\text{DST}_{i,j} + \beta\text{DDI}_{i,j}) \cdot \text{DSP}_i \tag{5}$$

where α, β are hyper-parameters that control the impact of $\text{DST}_{i,j}$ and $\text{DDI}_{i,j}$, respectively. Note that the formulation of Eq.5 is essentially an adjustment of the $\text{DST}_{i,j}$ score. Concretely, if c_i and c_j exhibits strong semantic dependency over extended contexts, it is expected that c_i and c_j are spatially far (i.e., with large $\text{DDI}_{i,j}$) and achieve a high $\text{DST}_{i,j}$, and $\text{DST}_{i,j}$ attributed to genuine semantic connections rather than repeated spans (as indicated by high DSP_i).

Totally, we define the long-dependency score for the entire sample S by accumulating $\text{LDS}_{i,j}$:

$$\text{LDS} = \sum_{i=1}^{N}\sum_{j=1}^{i-1} \text{LDS}_{i,j} \cdot I_{i,j} \tag{6}$$

$$I_{i,j} = \begin{cases} 1, & \text{if } \text{DST}_{i,j} > \tau \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where τ is a threshold for $\text{DST}_{i,j}$ to filter out noises in the perplexity calculation.

3.5 Enhancing Computational Efficiency

The time complexity for calculating LDS in Eq.6 is $O(N^2)$, which is impractical when handling massive candidate samples. To optimize the computational efficiency of LDS, we employ a method of random sampling. First, we randomly sample T index pairs: $\mathcal{D} = \{(x_1, y_1), ..., (x_T, y_T)\}$, where $x_t \in [1, N]$, $y_t \in [1, N]$ and $x_i < y_i$ for $t = 1, ..., T$. Each element $(x_t, y_t) \in \mathcal{D}$ corresponds to a segment pair (c_{x_t}, c_{y_t}). Then, a sampled LDS is calculated as:

$$\text{LDS}_{sp} = \sum_{(x_t,y_t)\in\mathcal{D}} \text{LDS}_{x_t,y_t} \cdot I_{x_t,y_t} \tag{8}$$

The computational of LDS_{sp} is substantially decreased, lowering the time complexity to $O(T)$.

4 Experimental Setup

4.1 Training Datasets

Our training datasets consist of three parts: pre-train dataset, English book dataset, and code dataset. The pre-training dataset is sampled from RedPajama (Computer, 2023) to mitigate catastrophic forgetting induced by extended training. The pre-training dataset part remains the same across all experiments. We choose English books and code data primarily for extending training because they naturally possess sufficiently long lengths. Note that the length of each document in all datasets exceeds 32k. Detailed training data information is provided in Appendix A.1.

4.2 Task Formulation

Assume that there are multiple models based on different training datasets and methods. We comprehensively evaluate the long context capability of the models, including three tasks as follows:

Language Modeling Tasks. Language modeling task is the most fundamental requirement for LLMs, typically measured by perplexity (PPL) on the test text data. We follow Chen et al. (2023b) to select 128 documents that are randomly sampled from the total proof-pile (Azerbayev et al., 2022) test split. For each document, it has at least 32768 tokens. All PPL results are calculated using the sliding window method (Press et al., 2021) with stride $S = 1024$.

Synthetic Long Context Tasks. The synthetic long context tasks are divided into two sub-tasks: multi-document question answering (MQA) and key-value retrieval. Following the methodology of Liu et al. (2023), we make controlled changes to the input context size and the position of the relevant information within this context in both sub-tasks. In the MQA task, the total number of documents is either 20 or 30, corresponding to text lengths of 4k or 6k tokens, respectively. For the key-value retrieval task, we use a total of 140 or 300 key-value pairs, corresponding to total token counts of 8k or 16k respectively. The goal of both task is to evaluate the model's ability to accurately identify unique relevant information from a large volume of irrelevant documents (or KV pairs).

Real-World Long Context Tasks. The current mainstream real-world long context tasks are derived from LongBench (Bai et al., 2023), primarily including single-document question answering, multi-document question answering, summarization, code completion, with the average length of most tasks ranging from 5k to 15k. We use the metrics and scripts provided along with the benchmark for evaluation.

4.3 Data Selection

In the experiments, we truncate the length M of all data to 32768 and set the segment length L to 128, thereby dividing each data instance into $N = 256$ segments. We set the hyper-parameters $\alpha = \beta = 1$ in LDS. To improve ProLong's computational efficiency, we replace the standard LDS with LDS_{sp}, utilizing a sampling size of $T = 5000$. We adopt a small model OPT-350m (Zhang et al., 2022) for calculating perplexity in LDS. We rank the book and code data based on their LDS, independently retaining those documents with high LDS from each source. These retained subsets, plus the pre-training dataset, are then integrated to construct our final training dataset. This method not only ensures the quality of our data but also enhances its diversity. In the experiments, we primarily compare the results of utilizing the entire dataset (Full), randomly selecting a 50% subset (Rand) and selecting the top-scoring 50% subset (ProLong) for extending training. We list the average LDS statistics under different data selection strategies in Appendix A.2.

4.4 Training Details

We extend the pre-trained Llama2-7b/13b (Touvron et al., 2023b) models to support context windows as large as 32768 tokens. We employ the NTK-aware (bloc97, 2023) method with base 160000 to rescale the position indices without any additional modifications to the Llama2 model architectures. Other hyper-parameters for training are displayed in Appendix B.1. For model comparison, we evaluate several mainstream LLMs with long context capability listed in Appendix B.2.

5 Experimental Results

5.1 ProLong Effectiveness

As shown in Table 1, we compare the long context performance of the models on different data selection methods (i.e., Full v.s. Rand 50% v.s. ProLong

Data Percentage	KV Retrieval (140 Pairs) Full (100%)	Rand (50%)	ProLong (50%)	KV Retrieval (300 Pairs) Full (100%)	Rand (50%)	ProLong (50%)	MQA (20 Documents) Full (100%)	Rand (50%)	ProLong (50%)	MQA (30 Documents) Full (100%)	Rand (50%)	ProLong (50%)
Llama2-7b	77.4	78.0	**93.4**	59.5	52.6	**86.0**	43.9	43.8	**45.2**	42.6	42.8	**44.4**
Llama2-13b	95.3	94.5	**95.4**	84.1	82.2	**84.1**	46.2	48.4	**49.1**	43.0	45.9	**46.1**

Data Percentage	HotpotQA Full (100%)	Rand (50%)	ProLong (50%)	2WikiMultihopQA Full (100%)	Rand (50%)	ProLong (50%)	MuSiQue Full (100%)	Rand (50%)	ProLong (50%)	GovReport Full (100%)	Rand (50%)	ProLong (50%)
Llama2-7b	44.4	42.4	**44.9**	**34.4**	33.6	34.2	**21.9**	19.9	21.3	30.2	29.4	**31.0**
Llama2-13b	**49.4**	48.2	48.6	38.5	38.2	**39.7**	19.1	17.0	**19.8**	32.0	32.1	**32.5**

Data Percentage	Qasper Full (100%)	Rand (50%)	ProLong (50%)	SAMSum Full (100%)	Rand (50%)	ProLong (50%)	LCC Full (100%)	Rand (50%)	ProLong (50%)	RepoBench-P Full (100%)	Rand (50%)	ProLong (50%)
Llama2-7b	**29.7**	27.6	28.3	42.7	42.4	**43.2**	64.9	64.4	**65.2**	58.5	59.2	**60.5**
Llama2-13b	38.4	36.9	**38.9**	43.1	43.2	**44.2**	67.5	66.9	**67.7**	**61.1**	60.5	60.8

Table 1: Results of ProLong on Llama-2-7B, Llama-2-13B. Full denotes the full dataset, and otherwise, we select 50% of the data with random selection (Rand). **Bold** numbers denotes the best-performing selected subset.

Model Setup	Multi-Document QA	
	20 Documents	30 Documents
ProLong-7b	**45.2**	**44.4**
w/o DSP	43.2 (-2.0)	43.1 (-1.3)
w/ DSP-add	43.3 (-1.9)	43.2 (-1.2)
w/o DDI	43.5 (-1.7)	43.8 (-0.6)
w/o DST	42.7 (-2.5)	42.8 (-1.6)

Table 2: Ablation study on the MQA task.

Model	Evaluation Context Window Size 2048	4096	8192	16384	32768
Llama2-7b	3.19	2.91	2.88	4.22	18.72
Code Llama-7b	3.53	3.17	2.93	2.77	2.84
Yarn-7b-64k	3.29	2.99	2.79	2.66	2.60
ProLong-7b	**3.02**	**2.76**	**2.58**	**2.45**	**2.38**
Llama2-13b	3.03	2.77	2.71	3.58	14.69
Code Llama-13b	3.40	3.06	2.82	2.67	2.71
Yarn-13b-64k	3.10	2.82	2.63	2.51	2.43
ProLong-13b	**2.89**	**2.64**	**2.47**	**2.35**	**2.28**

Table 3: Sliding window perplexity (stride S=1024) of 128 32k-length Proof-pile documents truncated to evaluation context window size.

50%). The evaluated benchmarks include KV retrieval, multi-document QA, and typical tasks in LongBench. We find that based on the same model (Llama2-7b or Llama2-13b), ProLong consistently outperforms random data selection across all the long context benchmarks. This result underscores the efficacy of ProLong in mining data with strong semantic dependencies across long contexts, which in turn enhances the long-context capabilities of LLMs. Interestingly, Table 1 also suggests that training with 50% of the data selected by ProLong often yields better results than training with the full dataset. This observation implies that most long contexts in the full dataset are not truly long at all. Surprisingly, the principle of "less is more" applies to certain specific long-context tasks.

5.2 Ablation Study on ProLong

As shown in Table 2, we also conduct ablation studies on multi-document question answering tasks. We first investigate the effect of the dependency specificity by removing it (denoted as "w/o DSP"), which is motivated to mitigate trivial dependencies introduced by repetitive patterns. The accuracy decreases in two different total document settings. Moreover, we observe a significant presence of repetitive patterns in the top-ranked data under the "w/o DSP" setting, with examples provided in Appendix E. Thus, the dependency specificity term indeed plays a crucial role in preventing the retrieval of meaningless repetitive patterns. Next, we examine the effect of using DSP as a multiplier instead of an addend in Eq. 6 (refer to Appendix D for the formula with DSP as an added bias, denoted as "w/ DSP-add"). The results indicate that using DSP as a bias is insufficient to mitigate the trivial dependencies introduced by repetitive patterns, and the performance of experiments is similar to that without DSP. Finally, we conduct experiments to analyze the impact of removing the dependency distance (denoted as "w/o DDI") or removing the dependency strength (denoted as "w/o DST") to do an ablation study. The results show that depen-

Model	140 Key-Value Pairs						300 Key-Value Pairs							
	p_1	p_{35}	p_{70}	p_{105}	p_{140}	Avg.	p_1	p_{50}	p_{100}	p_{150}	p_{200}	p_{250}	p_{300}	Avg.
GPT-4-32k	98.2	**98.2**	93.6	84.0	**100.0**	94.8	99.0	**94.0**	52.2	37.8	20.8	20.0	**99.8**	60.5
GPT-3.5-Turbo-16k	**100.0**	97.0	72.6	91.8	99.8	92.2	**100.0**	85.0	75.8	50.0	29.4	55.2	99.4	70.7
Yarn-7b-64k	97.6	19.2	11.0	29.0	92.8	49.9	71.6	10.8	1.6	1.0	4.6	3.8	73.2	23.8
Yarn-13b-64k	74.0	23.6	49.0	12.4	89.2	49.6	72.2	9.2	3.6	1.2	8.4	7.0	86.4	26.9
Vicuna-v1.5-13B-16k	97.8	79.2	84.6	78.6	28.2	73.7	-	-	-	-	-	-	-	-
ProLong-7b	99.6	95.2	91.8	91.6	88.8	93.4	98.0	61.8	**82.4**	**88.2**	86.4	93.4	92.0	**86.0**
ProLong-13b	**100.0**	96.2	**96.6**	**94.8**	89.4	**95.4**	99.8	86.0	73.6	59.4	**88.4**	**94.2**	87.6	84.1

Table 4: Key-value retrieval performance on dictionaries of 140 and 300 key-value pairs. p_i denotes the setting in which the relevant information is in the i-th position. "-" denotes that the result is not applicable due to exceeding context length after tokenization.

Model	Single-Doc QA		Multi-Doc QA		Summarization	Few-shot Learning	Code		Overall
	Narrative QA	Qasper	Hotpot QA	2WikiMulti hopQA	Gov Report	SAMSum	LCC	Repobench-p	All
GPT-3.5-Turbo-16k	23.6	43.3	51.6	37.7	29.5	41.7	54.7	53.6	42.0
Llama2-7b-chat-4k	18.7	19.2	25.4	32.8	27.3	40.7	52.4	43.8	32.5
LongChat-v1.5-7b-32k	16.9	27.7	31.5	20.6	30.8	34.2	53.0	55.3	33.8
Vincuna-v1.5-7b-16k	19.4	26.1	25.3	20.8	27.9	40.8	51.0	43.5	31.9
LongLora-7b-16k	19.8	29.1	37.0	30.3	**31.5**	41.9	57.6	54.5	37.7
Yarn-7b-64k	**25.0**	**30.8**	39.5	30.3	26.2	42.6	64.6	59.4	39.8
ProLong-7b	23.5	28.3	**44.9**	**34.2**	31.0	**43.2**	**65.2**	**60.5**	**41.4**
Llama2-13b-chat-4k	19.2	25.8	36.1	32.4	26.6	36.5	51.9	52.8	35.2
Vicuna-v1.5-13b-16k	18.9	29.9	38.1	36.0	27.9	27.8	44.1	45.6	33.5
PI-Llama2-13b-16k	19.2	33.3	44.9	34.8	27.9	27.9	62.5	51.1	37.7
Yarn-13b-64k	21.0	27.6	47.2	38.0	19.5	43.2	65.2	57.9	40.0
ProLong-13b	**23.0**	**38.9**	**48.6**	**39.7**	**32.5**	**44.2**	**67.7**	**60.8**	**44.4**

Table 5: Results (%) on single-doc QA, multi-doc QA, summarization, few-shot learning and code tasks from LongBench dataset. 'Overall' is computed by the macro-average over major task categories.

dency strength, as the core aspect, contributes the most to overall performance. Furthermore, removing DDI also leads to a decrease in performance. In conclusion, each term in the LDS calculation has its unique advantages and is indispensable in our ProLong framework. DST, DDI, and DSP collectively make the final LDS a feasible and effective index for mining data with long dependencies amidst a sea of long texts.

Given the effectiveness of ProLong, we train models from Llama2 (Touvron et al., 2023b) on the top of ProLong 50% (refer to Sec. 4.3 for setups in detail), resulting in **ProLong-7b/13b**.

5.3 Performance on Language Modeling

In Table 3, we compare our models ProLong-7b/13b with other baseline models in language modeling in various context window size, which is measured by perplexity. Firstly, ProLong-7b/13b exhibit a trend where perplexity decreases as the context window size increases, suggesting that they are capable of performing better language modeling with longer contexts. Additionally, under any context window size, our models based on ProLong significantly outperform other baseline models, including Llama2. This demonstrates that ProLong-7b/13b exhibits superior language modeling capabilities in long contexts.

5.4 Performance on Key Value Retrieval

We claim that lower PPL does not fully encapsulate the ability to handle long text tasks effectively. Thus, we also present the key value retrieval performance of ProLong-7b and ProLong-13b in Table 4. Compared with other models based on Llama2, ProLong-7b/13b surpasses them by a significant margin under the settings of 140 and 300 key-value pairs. Notably, ProLong-13b even outperforms the commercial model GPT-4-32k, especially in the more challenging setting of 300 key-value pairs. One key gap lies in that in the middle position, while other models show a significant drop in performance, our models based on the ProLong framework maintain high performance. This suggests

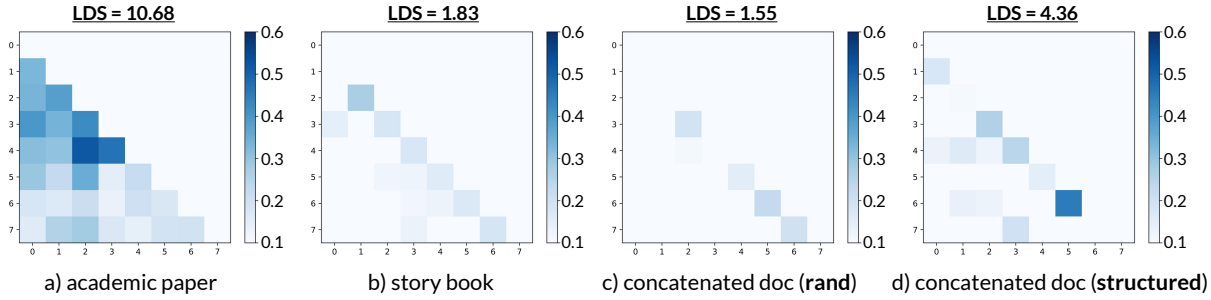

Figure 3: Visualization results of the dependency strength heat map for four data instances of equal length.

that the "lost in the middle" phenomenon, as described by Liu et al. (2023), is largely alleviated in our models. We also conducted experiments on the "needle in a haystack" task, which bears similarity to the key-value retrieval task. Detailed experimental results can be found in Appendix C.1. Consistent with the key-value retrieval task, the results demonstrate that the long context retrieval capability of ProLong-7b/13b experiences almost no degradation in the middle part of the context.

5.5 Performance on LongBench

For comprehensive evaluation on the long context performance, we further use real-world long context benchmarks (Bai et al., 2023). As shown in Table 5, we report the performance results (%) of partial English and code tasks on the LongBench dataset. Specifically, our ProLong-7b/13b achieves 41.1%/44.4% overall accuracy, which outperforms all the compared Llama2-based models by large margins. Besides, ProLong-13b surpasses the commercial model GPT-3.5-turbo-16k by an absolute gap of 2.4%. These results not only confirm the robust long-context capabilities of ProLong-7b/13b but also suggest that the ProLong framework holds promise for effectively addressing real-world long-context tasks. We also conducted experiments on standard short-length benchmarks, which further demonstrates ProLong can enhance the long-context modeling ability without compromising other capabilities. Detailed experimental results can be found in Appendix C.2.

6 Analysis

6.1 Balance between Accuracy and Efficiency

Initial LDS requires N^2 times precise calculations. In practice, sampling $T \ll N^2$ times and using a small model for perplexity calculations provides a more efficient, albeit approximate, solution. In this section, we explore the balance between accuracy and efficiency. We conduct experiments on the impact of different model sizes used for calculating perplexity (PPL) and different sampling granularity mentioned in Sec. 3.5. To assess the accuracy of ProLong's capability in selecting long-dependency data, We construct a toy test set from various sources. Specifically, we manually construct 100 instances with strong long-dependency as positive examples and 100 instances with weak long-dependency as negative examples. Subsequently, we use ProLong to score and rank the test set, assessing the number (accuracy) of positive examples contained in the top 100 ranked data.

# Samples	Model	Speed (documents/s)	Accuracy
5000	OPT-350m	0.16	89%
	Qwen-1.8b	0.04	89%
	Llama2-7b	0.01	96%
500	OPT-350m	**1.11**	87%
	Qwen-1.8b	0.32	87%
	Llama2-7b	0.08	96%

Table 6: Test result of ProLong by varying model sizes and sampling sizes.

We calculate the accuracy of ProLong in the test set to retrieve long-context documents and the number of documents processed per second as a measure of speed. The experimental results are as shown in Table 6, which reveals that smaller models and sampling sizes can achieve superior speed while sustaining comparable accuracy. In our experiments, we empirically choose OPT-350m and 5000 sampling size to perform ProLong, which may not necessarily yield the best results. Practically, it is essential to select an appropriate model and sampling size to strike a balance between accuracy and efficiency.

6.2 Visualization Analysis

To better illustrate the superiority of ProLong, we provide visualizations of dependency strength heat

maps for various documents under the same settings. Figure 3 displays various documents from the training set alongside their heat map visualization and LDS. LDS accurately captures the long-range dependencies within diverse documents. Specifically, academic papers typically showcase high LDS values, indicative of a more intricate heat map relationship and a pronounced presence of long-range dependencies. The same goes for and vice versa. Besides, we also find that the results of random concatenation of short texts (Figure 3.c) are very similar to those in storybooks (Figure 3.b), which demonstrates that even in naturally long texts, long-range dependencies may be minimal. Moreover, structured concatenated documents receive much higher LDS scores than randomly concatenated documents, corroborating part of the conclusions from the concurrent work, SPLICE (Staniszewski et al., 2023).

7 Conclusion

In this study, we propose ProLong, a novel and efficient approach for filtering long-dependency data in language model extension training. Benefiting from ProLong, we demonstrate that long-dependency data is the key to enhancing long-context modeling capability. Moreover, low-quality short-dependency data may even impair long-context modeling capability. Extensive experiments on multiple long-context benchmark datasets demonstrate the effectiveness of our ProLong in enhancing the long-context modeling capability of large language models. We hope ProLong can inspire researchers to spend more effort on data of more intrinsic long-dependency rather than long surface form.

Limitations

The ProLong framework can be used to identify documents that carry long dependencies effectively, and LLM trained on these documents exhibit significantly enhanced long-context modeling capabilities. However, to build strong LLMs, ProLong should be used in combination with other techniques. Here we list some of the limitations that are not considered when designing ProLong: (1) Diversity of Corpora. Our experiment only considers English books and code texts. In future work, we should extend our analysis to cover a wider range of long-context data, including different languages and document types. (2) Model Considerations. In this study, our experiments are solely performed on Llama2-7B and Llama2-13B models. It is expected to obtain better performance with larger base models. (3) Data Mixture Percentage. We do not perform comprehensive exploration of the data mixture percentage, which is reported to be important to the final performance of LLM.

In future studies, we hope to explore the above-listed limitations further.

Ethics Statement

Our study does not carry any ethical concerns. Specifically, our training data are publicly available and designated for research purposes only. We inspect our dataset to ensure it does not contain any unethical content, private information and offensive topics. Moreover, the base models we used are also publicly available for research purposes.

Acknowledgments

Min Yang was supported by National Key Research and Development Program of China (2022YFF0902100), National Natural Science Foundation of China (62376262), the Natural Science Foundation of Guangdong Province of China (2024A1515030166), Shenzhen Science and Technology Innovation Program (KQTD20190929172835662), Shenzhen Basic Research Foundation (JCYJ20210324115614039).

References

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

Zhangir Azerbayev, Edward Ayers, and Bartosz Piotrowski. 2022. Proof-pile.

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

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open llm leaderboard. `https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard`.

bloc97. 2023. Ntk-aware scaled rope allows llama models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

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

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

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023b. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*.

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. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Together Computer. 2023. Redpajama: an open dataset for training large language models.

Harm de Vries. 2023. In the long (context) run.

Yao Fu, Xinyao Niu, Xiang Yue, Rameswar Panda, Kim, Yoon, and Hao Peng. 2023. Understanding data influence on context scaling. *Yao Fu's Notion*.

Daya Guo, Canwen Xu, Nan Duan, Jian Yin, and Julian McAuley. 2023. Longcoder: A long-range pre-trained language model for code completion. *arXiv preprint arXiv:2306.14893*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

He Junqing, Pan Kunhao, Dong Xiaoqun, Song Zhuoyang, Liu Yibo, Liang Yuxin, Wang Hao, Sun Qianguo, Zhang Songxin, Xie Zejian, et al. 2023. Never lost in the middle: Improving large language models via attention strengthening question answering. *arXiv preprint arXiv:2311.09198*.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2021. Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499*.

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?

Yunshui Li, Binyuan Hui, Xiaobo Xia, Jiaxi Yang, Min Yang, Lei Zhang, Shuzheng Si, Junhao Liu, Tongliang Liu, Fei Huang, et al. 2023b. One shot learning as instruction data prospector for large language models. *arXiv preprint arXiv:2312.10302*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

OpenAI. 2022. Introducing chatgpt.

Arka Pal, Deep Karkhanis, Manley Roberts, Samuel Dooley, Arvind Sundararajan, and Siddartha Naidu. 2023. Giraffe: Adventures in expanding context lengths in llms. *arXiv preprint arXiv:2308.10882*.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*.

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

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Konrad Staniszewski, Szymon Tworkowski, Sebastian Jaszczur, Henryk Michalewski, Łukasz Kuciński, and Piotr Miłoś. 2023. Structured packing in llm training improves long context utilization. *arXiv preprint arXiv:2312.17296*.

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

Simeng Sun, Kalpesh Krishna, Andrew Mattarella-Micke, and Mohit Iyyer. 2021. Do long-range language models actually use long-range context? *arXiv preprint arXiv:2109.09115*.

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

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

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. 2023. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*.

Yijiong Yu. 2023. " paraphrasing the original text" makes high accuracy long-context qa. *arXiv preprint arXiv:2312.11193*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Lei Zhang, Yunshui Li, Ziqiang Liu, Junhao Liu, Min Yang, et al. 2023. Marathon: A race through the realm of long context with large language models. *arXiv preprint arXiv:2312.09542*.

Peitian Zhang, Zheng Liu, Shitao Xiao, Ninglu Shao, Qiwei Ye, and Zhicheng Dou. 2024. Soaring from 4k to 400k: Extending llm's context with activation beacon. *arXiv preprint arXiv:2401.03462*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, and Yanlin Wang. 2023. Memorybank: Enhancing large language models with long-term memory. *arXiv preprint arXiv:2305.10250*.

Yucheng Zhou, Tao Shen, Xiubo Geng, Chongyang Tao, Guodong Long, Can Xu, and Daxin Jiang. 2022. Fine-grained distillation for long document retrieval. *arXiv preprint arXiv:2212.10423*.

A Training Datasets

A.1 Training Datasets Overview

All training corpus we used in our experiments are given in Table 7. The majority of the data consists of English books and code, all of which have lengths exceeding 32k.

Type	Dataset	# Instance	# Len
Pre-train Data	RedPajama	98k	1.2k
English Book	arXiv	200k	54.7k
	Book3	279k	165.2k
	BookCorpus2	43k	103.6k
	PG19	20k	137.4k
Code	C	40k	108.3k
	C++	40k	105.7k
	Java	40k	94.2k
	Python	40k	87.9k

Table 7: Training datasets overview. # Instance represents the number of instances, and # Len represents the average length of each instance in the dataset.

A.2 Average Long-Dependency Score

In the experiments, we primarily compare the results of utilizing the entire dataset (Full), randomly selecting a 50% subset (Rand), and selecting the top-scoring 50% subset (ProLong) for extending training. We analyze the average long-dependency scores in these three scenarios, as detailed in Table 8, where the results for Rand and Full are very close across different data sources, but the results for ProLong are significantly higher than both of them.

Data Type	Datasets	Average LDS		
		Full (100%)	**Rand** (50%)	**ProLong** (50%)
English Book	arXiv	1418.0	1418.0	1890.3
	Book3	385.4	386.5	1028.9
	BookCorpus2	327.2	326.2	544.1
	PG19	469.9	467.3	1202.9
Code	C	796.1	798.4	2484.0
	C++	1239.3	1240.0	2909.6
	Java	1219.9	1219.2	2234.1
	Python	828.7	824.2	1758.8

Table 8: The average LDS under different data selection strategies on the training datasets.

B Experimental Details

B.1 Training Hyper-parameters

All model variants are trained via the next token prediction objective. We set a learning rate of 2×10^{-5} with no weight decay and the whole training step is set to 6000 with a global batch size of 128. For subsequent experiments with 50% data setting, the training step is set to 3000 steps. Additionally, we adopt a linear learning rate warm-up scheduler over 20 steps and the AdamW (Loshchilov and Hutter, 2017) optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.95$.

B.2 Model Baselines

We evaluate several mainstream LLMs with long context capability, including GPT-4-32k (Achiam et al., 2023), GPT-3.5-Turbo-16k (OpenAI, 2022), Llama2-7b (Touvron et al., 2023b), Llama-13b, Code Llama-7b (Roziere et al., 2023), Yarn-7b-64k (Peng et al., 2023), Yarn-13b-64k, Llama2-7B-chat-4k (Touvron et al., 2023b), Llama2-13B-chat-4k, LongChat-v1.5-7B-32k (Li et al., 2023a), Vicuna-v1.5-7B-16k (Chiang et al., 2023), Vicuna-v1.5-13B-16k, Longlora-7B-16k (Chen et al., 2023b), and PI-Llama2-13B-16k (Chen et al., 2023a).

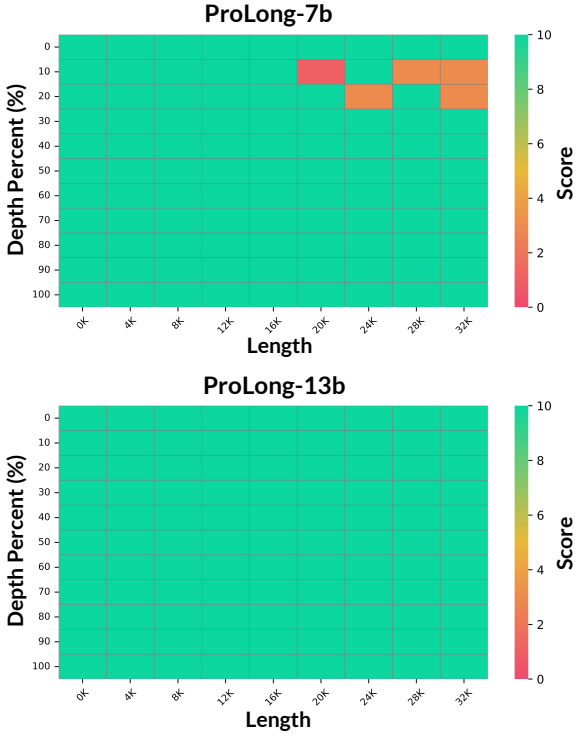

Figure 4: 0k-32k pressure test (Needle In A HayStack) performance of ProLong-7B and ProLong-13B.

Model	ARC-c	Hellaswag	MMLU	TruthfulQA	Winogrande	GSM8K	**Avg.**
Llama2-7b	0.52	0.79	0.46	0.39	0.69	0.13	0.50
ProLong-7b	0.54	0.79	0.45	0.40	0.68	0.14	0.50
Llama2-13b	0.60	0.82	0.55	0.37	0.72	0.23	0.55
ProLong-13b	0.60	0.82	0.53	0.38	0.72	0.24	0.55

Table 9: Performance on a subset of standard short-length benchmarks.

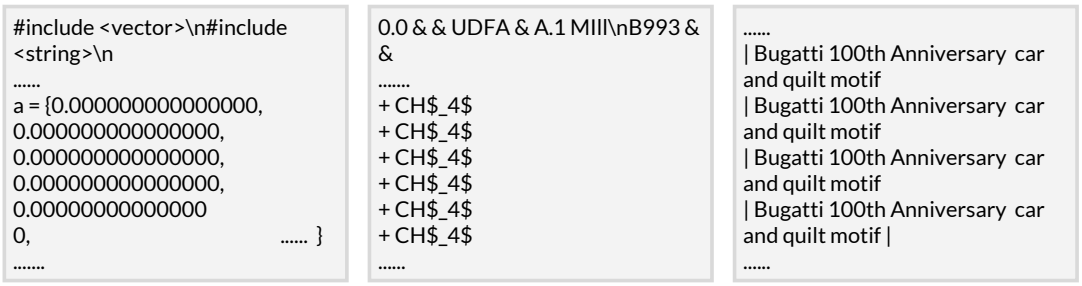

Figure 5: Repetitive patterns examples from arXiv, C++, PG19 datasets.

C More Experimental Results

C.1 Needle in A HayStack Task

A simple 'needle in a haystack' analysis to test in-context retrieval ability of long context LLMs. This test involves placing a random fact or statement (the 'needle') in the middle of a long context window (the 'haystack'), requiring the model to retrieve this needle. Then, it iterates over various document depths (where the needle is placed) and context lengths to measure performance. In the setting where the maximum context length is 32k, the results in Figure 4 show ProLong-13B obtains 100% needle in a haystack accuracy across all tested depths and context lengths. ProLong-7B exhibits a few errors when the context length exceeded 16k, it still maintains an overall accuracy of 95%.

C.2 Standard short-length benchmarks

As shown in the table 9, we evaluate ProLong-7b and ProLong-13b on several standard short-length benchmarks from Hugging Face Open LLM Leaderboard (Beeching et al., 2023), i.e., 25-shot ARC-Challenge (Clark et al., 2018), 10-shot Hellaswag (Zellers et al., 2019), 5-shot MMLU (Hendrycks et al., 2020), 0-shot TruthfulQA (Lin et al., 2021), 5-shot Winogrande (Sakaguchi et al., 2021), 5-shot GSM8K (Cobbe et al., 2021). When compared with the baseline Llama2-7b and Llama2-13b models, we find that ProLong does not exhibit a significant performance decline, on the contrary, it achieves comparable or even better performance than baseline.

D Additive Dependency Specificity

Additive Dependency Specificity is calculated as follows:

$$\mathrm{LDS} = \sum_{i=1}^{N} \sum_{j=1}^{i-1} \{ (\alpha \mathrm{DST}_{i,j} + \beta \mathrm{DDI}_{i,j} + \gamma \mathrm{DSP}_i) \times I_{i,j} \} \quad (9)$$

where γ is a hyper-parameters. In the ablation experiments, we set the hyper-parameters $\alpha = \beta = \gamma = 1$ in LDS.

E Repetitive Patterns

As shown in Figure 5, we present some examples of repetitive patterns in the training data. To be specific, the repetitive patterns in C++ are a huge number of reptitive numbers, in the arXiv are numerous repetitive formula symbols, and in the PG19 are an abundance of reptitive tables, which may potentially have a negative impact. When we do not use dependency specificity or use additive dependency specificity, this type of data will receive a high long-dependency score, thereby impacting the quality of the filtered data.