# Revisiting Long-context Modeling from Context Denoising Perspective

**Anonymous ACL submission**

## Abstract

Long-context models (LCMs) have demonstrated great potential in processing long sequences, facilitating many real-world applications. The success of LCMs can be attributed to their ability to locate implicit critical information within the context for further prediction. However, recent studies indicate that LCMs can be distracted by the context noise (irrelevant information). In this paper, we conduct a fine-grained analysis of the context noise and propose an effective metric, i.e., IG score, for identifying noise information within the context. We also find that simply restraining the effect of noisy context can significantly boost the model's attention on critical tokens. Based on this observation, we propose a simple yet effective training strategy, CDT (Context Denoising Training), which can simultaneously strengthen the model's attention on critical tokens and achieve a stronger connection between these critical tokens and the model prediction. Experiments on both context window scaling and long-context alignment settings across 4 different tasks exhibit the superiority of CDT. With CDT, an open-source 8B model can achieve results (50.92 points) comparable to GPT4o (51.00 points)[1].

## 1 Introduction

The ability to handle long input sequences has become a fundamental requirement for large language models (LLMs), with few cutting-edge models capable of processing context lengths exceeding millions of tokens (Team et al., 2024; MiniMax et al., 2025). This advancement eliminates the need for complex toolchains and intricate workflows, e.g., RAG (Yu et al., 2024), and significantly enhances real-world applications, such as long-document summarization (Laban et al., 2024) and project code analysis (Fang et al., 2024a).

[1]Our code is available at https://anonymous.4open.science/r/context-denoising-training-D7DF

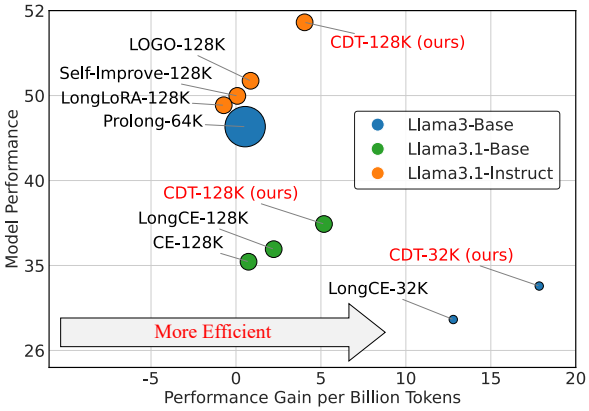



Figure 1: Comparative overview of *model performance* on real-world long-context tasks and *performance gain per billion tokens* among different training methods. The bubble size indicates the training data volume.

Yet, recent studies have indicated that LCMs frequently fail when handling with long-context tasks (Hsieh et al., 2024; Kuratov et al., 2024; Tang et al., 2024b; Bai et al., 2024c), and the open-source community mitigates such an issue mainly by using sufficient high-quality synthetic long-context data to post-train the model (Fu et al., 2024a; Chen et al., 2024; Gao et al., 2024a). However, these approaches are proven either inefficient or ineffective under limited resources. For example, as shown in Figure 1, Prolong-64K-Base (Gao et al., 2024b) achieves significant performance but improves by only 0.3 points per 1B tokens used. In contrast, LongCE (Fang et al., 2024b) exhibits less improvement but achieves nearly 13 points per 1B tokens, demonstrating significantly higher training efficiency. One of the possible reasons is that existing works overlook the fact that LCMs process long input in a *retrieval-then-generation* manner, i.e., first implicitly identifying key information within the context and then performing generation with the aggregated context (Liu et al., 2024b; Wu et al., 2024; Li et al., 2024a; Qiu et al., 2025), but the critical tokens in the "retrieved-context" might be

overwhelmed by excessive irrelevant context (Ye et al., 2024). Thus, the key to achieving better long-context modeling is *effectively detecting the critical tokens and diminishing the effect of context noise (irrelevant tokens)*.

In this paper, we conduct a fine-grained analysis to investigate the impact of context noise on long-context modeling. Specifically, we propose a novel critical token detection metric, the IG score, based on the concept of information flow (Wang et al., 2023). Our approach achieves a remarkable accuracy improvement in the critical token detection task compared to traditional attention-based metrics. Then, we manually diminish context noise by subtracting the gradients of token embeddings in irrelevant contexts. We find that simply suppressing context noise at the model input allows LCMs to focus more effectively on critical tokens.

Built upon the above analysis, we propose a simple yet effective Context Denoising Training (CDT) strategy, which performs denoising at the model input, allowing the model to focus more effectively on critical tokens to better establish the connection between critical tokens and generation. Notably, our CDT approach is analogous to the Signal Denoising in the digital signal processing field (Kopsinis and McLaughlin, 2009), where noise reduction in the input sequence can enhance the model's attention to essential parts within the context. Experiments on two essential long-context training scenarios, i.e., context window scaling and long-context alignments, across 4 different types of long-context tasks (real-world tasks, language modeling, synthetic tasks, and long-form reasoning tasks) exhibit the superiority of our method. Our CDT can consistently surpass the other methods with an average gain of 2 points on 12 real-world long-context tasks in LongBench (Bai et al., 2024b) and 13 long synthetic tasks in RULER (Hsieh et al., 2024). Additionally, with CDT, an open-source 8B model can achieve comparable results with GPT4o on real-world tasks (50.92 points v.s. 51.00 points).

## 2 Related Work

### 2.1 Long-context Modeling

Generally, the purposes of long-context modeling can be categorized into two types: context window scaling and long-context alignment. For context window scaling, many works have explored to post-train the LLMs with minimal post-training, including position extrapolation (Chen et al., 2023a; Peng et al., 2023; Ding et al., 2024; Liu et al., 2024a; Zhao et al., 2024a; Zhang et al., 2024c; Fu et al., 2024b; Lu et al., 2024) and modifying the attention pattern (Chevalier et al., 2023; Chen et al., 2023b; Xiao et al., 2024b; Bertsch et al., 2024). Another line of work focuses on enhancing models that already possess long-context window, aiming to help LCMs capture critical information within the context (Liu et al., 2024b; An et al., 2024; Gao et al., 2024c; Xiong et al., 2024) and reducing misalignment issues like hallucinations (Zhang et al., 2024b; Tang et al., 2024a; Li et al., 2024b). In this work, we rethink the long-context training from the context denoising perspective and propose a CDT strategy, aiming to improve training effectiveness.

### 2.2 Retrieval-then-generation Mechanism of Long-context Models

Existing research has demonstrated that LCMs handle long-context in a retrieval-then-generation manner, where LCMs first retrieve salient information within the context and utilize these information to generate responses (Wu et al., 2024; Tang et al., 2024b; Zhao et al., 2024b; Qiu et al., 2025). However, Liu et al. (2024b) observe the "lost-in-the-middle" phenomenon of LCMs, which highlights that LCMs exhibit a positional bias toward locating key information. Furthermore, Ye et al. (2024) and Fang et al. (2024b) discover that excessive irrelevant long-context can overwhelm critical information, thereby impairing the performance of the model. To mitigate the above issue, some works have explored solutions from various perspectives, including model architecture improvements (Ye et al., 2024; Xiao et al., 2024a), enhancements in information extraction mechanisms (Li et al., 2024a; Zhang et al., 2024a), and optimization of training objective (Fang et al., 2024b; Bai et al., 2024a). In this paper, we revisit critical information location from the context denoising aspect, helping model establish better connections between salient tokens and generation by achieving more accurate identifying and effective diminishing of noise input.

## 3 Preliminary Study

In this section, we analyze the influence of context noise, i.e., irrelevant tokens, on long-context modeling. More concretely, we first design critical token detection metrics in §3.1 and study the impact of context noise restraint on long-context modeling in §3.2. We conduct experiments with the

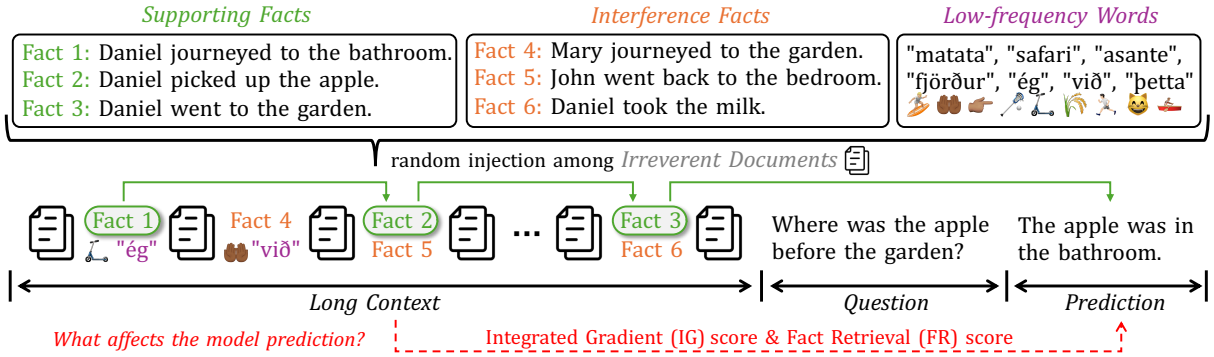



Figure 2: Task format of our preliminary study, which requires models to predict the final answer by reasoning through multi-hop Supporting Facts and distinguishing from the Interference Facts. Simultaneously, the model should resist the influence of Texts from Books and Low-Frequency Words. More details are shown in Appendix A.

Llama3.1-8B-Instruct (Meta, 2024) model, which owns a 128K context window size.

**Synthetic Task Format** We design a synthetic long-form reasoning task for the following studies. As shown in Figure 2, there are four types of tokens in our task: supporting facts, interference facts, low-frequency words, and texts from books. LCMs are tasked with predicting the correct answer (e.g., "bathroom") by reasoning through multiple supporting facts within the long context. Notably, the interference facts are seemingly related to the answers and are randomly inserted into the context, aiming to distract the models from providing the correct response. Therefore, LCMs should predict based on *critical tokens*[2] while also preventing these two types of tokens from being overwhelmed by *irrelevant tokens*, including excessive irrelevant documents and low-frequency words.

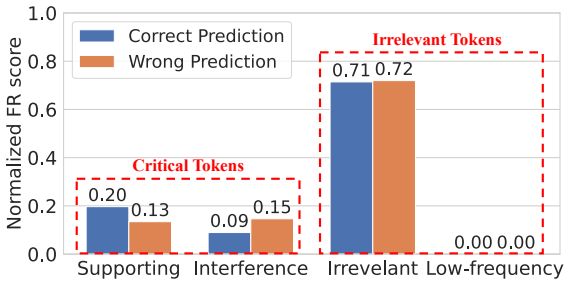
### 3.1 Critical Tokens Detection

We start by comparing two metrics: FR score and IG score, on the critical tokens (including both supporting and interference facts) detection task. Given the input sequence $X = \{x_i\}_{i=1}^{n}$ and the ground truth $Y = \{y_j\}_{j=1}^{m}$, we define FR score and IG score as follows:

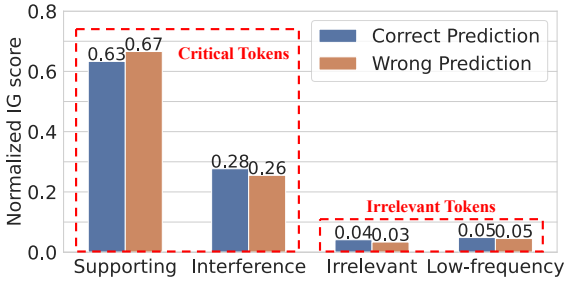
**Attention Distribution Metric: FR score** Existing works primarily identify critical tokens based on the attention distribution (Wu et al., 2024; Gema et al., 2024; Xiao et al., 2024a). Similarly, we design the Fact Retrieval (FR) score for our synthetic task based on the attention distribution to quantify the model's attention allocated to different types of tokens. At each step of model prediction $y_j$, if the attention score of $x_i$ ranks within the top-k across the entire sequence, we define $x_i$ as being attended by an attention head. Let $s_j$ be the set of tokens attended by an attention head at the generation step $j$, and $\mathcal{T}_r$ refers to the context token set of type $r \in \{\text{sup}, \text{inter}, \text{irr}, \text{low}\}$, e.g., $\mathcal{T}_{sup}$ denotes tokens of the supporting facts. The FR score $\text{FR}_{h,l}^{(r)}$ of the $h$-th attention head in the $l$-th model layer can be written as:

$$\text{FR}_{h,l}^{(r)} = \frac{|\ s_j \cap \mathcal{T}_r\ |}{|\ \mathcal{T}_r\ |}.$$

We average FR scores from all heads to reflect the attention distribution of tokens in $\mathcal{T}_r$.




(a) Attention distribution reflected by average FR score.




(b) Information flow reflected by average IG score.

Figure 3: Comparison between attention distribution and information flow on critical token location task.

[2]Since these tokens are highly likely to be correlated with the answers. However, LCMs should distinguish between supporting facts and interference facts to predict accurately.

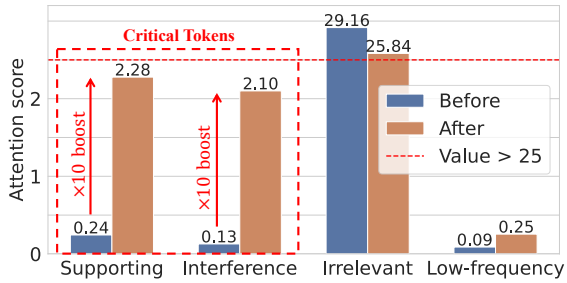



Figure 4: Attention distributions before and after manual context denoising. After context denoising, attention scores on critical tokens boost $\times 10$ times.

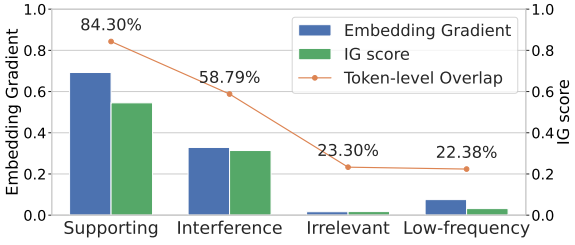



Figure 5: Relationship between attention IG score and L2 normalized embedding gradients on different types of tokens. It shows a proportional correlation.

**Information Flow Metric: IG score** To discover the attention interaction among tokens, i.e., information flow, we employ the integrated gradient (IG) technique (Wang et al., 2023) on the attention module. The IG score on the attention module from the $l$-th model layer can be defined as:

$$\mathrm{IG}_l = \sum_h \mid A^T_{h,l} \odot \frac{\partial \mathcal{L}_\theta(Y|X)}{\partial A_{h,l}} \mid, \qquad (1)$$

where $\mathcal{L}_\theta(Y|X)$ is the model prediction loss. We calculate IG scores between each $x_i \in X$ and $y_j \in Y$, i.e., $\sum_j \mathrm{IG}_l(i,j)$, and average these scores from all attention heads. A higher average IG score indicates a larger contribution from $x_i$ to $Y$.

**Observation** For a clear comparison, we normalize the computed FR and IG scores, and plot them in Figure 3. We find that the IG score detects significantly less noise (irrelevant documents and low-frequency tokens) compared to the FR score on critical token detection. Specifically, as shown in Figure 3a, attention-based metrics reflect the distribution of tokens that the model focuses on during the generation process. When the model generates correct responses, its attention focuses more on supporting facts; when the model generates wrong responses, its attention focuses more on interference tokens. Yet, in both cases, the FR score indicates that the model significantly focuses on irrelevant tokens. As for the IG score shown in Figure 3b, it reflects the contribution of each token to the final prediction based on the loss computed on generated results. Regardless of whether the response is correct or not, the IG score for critical tokens is significantly higher than that for irrelevant tokens. Therefore, we can effectively identify the critical tokens by leveraging the IG score and establish a better connection between critical tokens and generation through subsequent training.

### 3.2 Effect of Context Noise Restraint

Considering that directly suppressing context noise in attention is very challenging, we aim to restrain the noise from the input perspective. Given the positions of irrelevant tokens, we subtract their token embedding gradients to suppress the noise. This is motivated by the fact that the model has largely converged on these noisy tokens, resulting in their gradients exhibiting low sensitivity. As shown in Figure 4, we observe that after manual context denoising, the attention scores on critical tokens increase nearly $\times 10$ times, while the attention scores on irrelevant contextual tokens exhibit a slight decrease. It is worth noting that this operation can be analogized to *denoising in the digital signal processing field* (Kopsinis and McLaughlin, 2009), as it reduces noise in the input sequence, allowing the model to focus more effectively on the under-fitting critical tokens.

## 4 Context Denoising Training

Based on the above observation, we propose a simple yet effective Context Denoising Training (**CDT**) strategy, which suppresses context noise during training to strengthen the model's attention on critical tokens and help establish a better connection between critical tokens and the final prediction. CDT involves two key steps: (1) Critical Token Detection and (2) Emphasizing Training.

### 4.1 Critical Token Detection

Intuitively, we can first apply IG score to detect the critical tokens for the subsequent training. However, computing the IG score in long-context scenario is highly GPU memory-intensive, as it requires storing full attention gradients and weights from every model layer across the entire sequence. *Even with 8×92GB GPUs (H20), the maximum computable sequence length for the IG score is lim-*

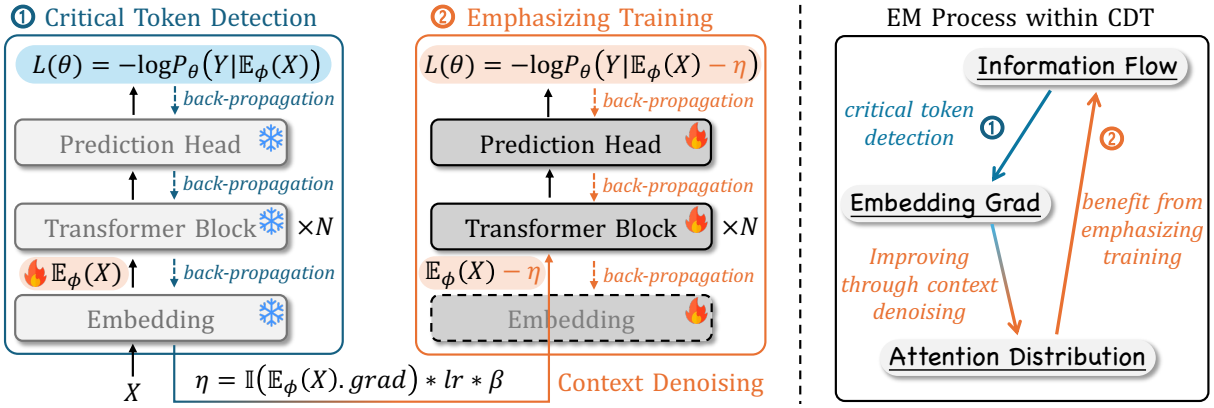



Figure 6: Our proposed CDT (context denoising training) method. It consists of two steps: (1) detecting critical tokens within the long context, and (2) utilizing the denoised context for further emphasizing training. Notably, CDT can be understood as an *Expectation Maximization (EM)* process, where the model detects noise based on information flow and improves the training by diminishing the noise, thereby enhancing the information flow.

*ited to 12K*, making it infeasible to generalize to a longer sequence. Therefore, we designed a simple alternative implementation, which approximates the IG score with token embedding gradients[3]. We derive a proportional relationship between the token embedding gradient and the IG score, and visualize the results in Figure 5. A detailed derivation is provided in Appendix B. As shown in Figure 6, given the input sequence $X = \{x_i\}_{i=1}^n$, label $Y$, and the model $f_\theta$, we first freeze the model parameters, keeping only the gradients of the input token embeddings $E_\phi(X)$, where $\phi \subset \theta$. We then obtain the gradient of each token embedding through the computation of the cross-entropy (CE) loss followed by a loss back-propagation. To identify the critical tokens, we employ an identifier $\mathbb{I}(\cdot)$ to detect tokens with large gradients, i.e., critical tokens, in the sequence. Specifically, we define the calculation of the significance of each token as comparing its L2-normalized embedding gradient against the average of the computed gradients of all tokens. This detection process can be written as:

$$t = \frac{1}{n}\sum_{i=1}^{n} ||\nabla_{E_\phi(x_i)} \mathcal{L}_{\text{CE}}(x_i)||_2,$$

$$\mathbb{I}(x_i) = \begin{cases} 0, & \text{if } ||\nabla_{E_\phi(x_i)} \mathcal{L}_{\text{CE}}(x_i)||_2 < t, \\ 1, & \text{if } ||\nabla_{E_\phi(x_i)} \mathcal{L}_{\text{CE}}(x_i)||_2 \geq t, \end{cases}$$

where $\mathbb{I}(x_i) = 1$ means $x_i$ is the critical token; otherwise, it is irrelevant context (noise).

[3]We choose token embeddings for three main reasons: (1) they are easily accessible, (2) the gradients of embeddings are directly associated with each token, and (3) they require significantly less GPU memory compared to attention gradients.

### 4.2 Emphasizing Training

To suppress the context noise, we leverage the computed gradients to manipulate the embeddings of irrelevant tokens, leaving critical tokens unchanged. More concretely, each irrelevant token embedding can be denoised as:

$$E_\phi(x_i)' = E_\phi(x_i) - \mathbb{I}(x_i)\nabla_{E_\phi(x_i)} \times lr \times \beta, \quad (2)$$

where $lr$ is the learning rate and $\beta$ is the hyper-parameter controlling the denoising level. Then, we unfreeze the model and use the denoised token embeddings as the model input for further training, which we refer to as Emphasizing Training. The loss function can be formulated as:

$$\mathcal{L}_{CDT}(X, Y) = \mathcal{L}_{CE}\left(f_\theta\left(E_\phi(X)'\right), Y\right). \quad (3)$$

**Remark** Notably, the above process is conducted online during training rather than pre-computed offline. As shown in Figure 6, although this introduces additional computational overhead, CDT bootstraps the model's long-context capabilities in an *Expectation-Maximization (EM) manner*: the model first identifies the critical tokens based on information flow and improves the training by diminishing the noise, thereby ultimately enhancing the information flow. In § 6.3, we will demonstrate that, by training with CDT, the model can continuously enhance its performance compared to conventional training objectives.

## 5 Experiment

### 5.1 Settings

**Evaluation** We evaluate the model performance on 4 different types of long-context tasks, includ-

| Models | Type | S-Doc QA | M-Doc QA | Summ | Few-shot | Code | Avg. |
|---|---|---|---|---|---|---|---|
| ProLong-512K-Instruct (Gao et al., 2024b) | SFT | 40.07 | 41.24 | 28.27 | 64.21 | 63.08 | 47.37 |
| NExtLong-512K-Instruct (Gao et al., 2025) | SFT | 43.47 | 43.21 | 29.88 | 60.87 | 44.35 | 44.35 |
| Llama-3.1-8B-SEALONG (Li et al., 2024b) | DPO | 49.45 | 44.69 | **30.96** | 61.54 | 57.85 | 48.90 |
| GPT-4o (version: 2024-11-20) | - | **51.43** | **60.89** | 14.78 | **66.37** | **61.25** | **51.00** |
| **Results on Short-context Model** (*all SCMs share the same training data, 8× context window scaling.*) | | | | | | | |
| Llama-3-8B-Base (8K) | - | 25.20 | 21.52 | 20.18 | 32.67 | 27.92 | 25.50 |
| + YaRN (Peng et al., 2023) | - | 24.37 | 19.86 | 24.32 | 29.99 | 31.67 | 26.04 |
| + CE | CWS | **25.29** | 21.49 | 20.36 | 32.69 | 27.76 | 34.62 |
| + LongCE (Fang et al., 2024b) | CWS | 17.13 | 9.59 | 25.00 | 59.57 | 61.83 | 34.62 |
| + CDT (ours) | CWS | 17.03 | **24.87** | **26.61** | **61.89** | **66.14** | **39.31** |
| **Results on Long-context Base Model** (*all LCMs share the same training data.*) | | | | | | | |
| Llama-3.1-8B-Base | - | 18.20 | **13.19** | 21.13 | **63.80** | 69.32 | 37.13 |
| + CE | LM | 17.10 | 10.82 | 26.38 | 62.85 | **70.62** | 37.55 |
| + LongCE (Fang et al., 2024b) | LM | 19.14 | 10.87 | 28.63 | 59.63 | 66.24 | 36.90 |
| + CDT (ours) | LM | **19.15** | 13.01 | **29.23** | 63.63 | 69.44 | **38.89** |
| **Results on Long-context Instruct Model** (*all LCMs use same source data with different formats.*) | | | | | | | |
| Llama-3.1-8B-Instruct | - | 48.58 | 45.19 | 30.30 | 61.73 | 57.26 | 48.61 |
| + SFT | SFT | 49.23 | 44.86 | 30.39 | 61.96 | 57.14 | 48.72 |
| + LOGO (Tang et al., 2024a) | DPO | 49.63 | 45.39 | **30.44** | 62.39 | 57.19 | 49.01 |
| + CDT (ours) | SFT | **50.11** | **46.04** | 30.34 | **62.49** | **65.64** | **50.92** |

Table 1: Evaluation results on LongBench-E benchmark. *To ensure comparison fairness, we place existing works that do not use the same training data with us in the top group*. We implement our method under different settings, including context-window scaling (CWS), language modeling (LM), SFT, and DPO.

ing real-world tasks (LongBench-E (Bai et al., 2024b), language modeling task (LongPPL (Fang et al., 2024b)), long-form reasoning task (Babilong (Kuratov et al., 2024)), and synthetic tasks (RULER (Hsieh et al., 2024)). We compare CDT against existing widely-used methods on two types of models: (1) short-context models (SCMs) that require *context window scaling*; (2) long-context models (LCMs) that require *long-context alignment*. In our main experiments, we select Llama-3-8B-Base model as the SCM, of which context window size is scaled $\times 8$ times. For LCMs, we select Llama-3.1-8B-Base as LCM-Base and Llama-3.1-8B-Instruct model as LCM-Instruct. We provide more evaluation details in Appendix C, and show more evaluation results such as generalizing to more models in Appendix D.

**Dataset Construction and Training Details** For context window scaling training on SCM and post-training on LCM-Base, we apply PG-19 (Rae et al., 2019) as the training data. For each training sample, we organize it into 64K tokens and collect 10,000 training samples. For long-context alignment on LCM-Instruct, we post-process the data sampled from LongMiT (Chen et al., 2024) and LongAlpaca (Chen et al., 2023c), two publicly available long-context QA datasets. Finally, it contains 8,000 samples for long-context alignment training, covering context lengths from 16K to 128K. Empirically, we set $\beta = 5$ in Equation 2 in all experiments. More dataset processing and implementation details are shown in Appendix C

### 5.2 Results

**Real-world Long-context Understanding Tasks** LongBench-E is a comprehensive benchmark suite encompassing 12 real-world datasets and various context lengths spread across 5 categories, including Single Document QA (S-Doc QA), Multi-Document QA (M-Doc QA), Summarization (Summ), Few-shot, and Code. As shown in table 1, we can observe that: **(1)** *CDT achieves the best performance among all the sub-tasks*. For SCMs, with the same training data, CDT achieved the best performance, outperforming a competitive counterpart (LongCE) by nearly 4.7 points on average. **(2)** For LCM-Base models, we find when post-training on the base model with language modeling training objective, *CDT is the only method that ensures no significant performance drop across all subtasks*, and it even achieves some improvements. In con-

| Models | RULER | | Language Modeling | | BABILong | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 32K | 64K | LongPPL | PPL | 4K | 8k | 16k | 32k | 64k | Avg. |
| Llama-3-8B-Base | 0 | 0 | > 100 | > 100 | 33.40 | 26.60 | 4.80 | 0.00 | 0.20 | 13.00 |
| + YaRN | 39.58 | 31.46 | 3.55 | 5.60 | 35.20 | 29.80 | 24.40 | 20.20 | 17.60 | 25.44 |
| + CE | 36.01 | 13.82 | 3.90 | 6.46 | 36.60 | 34.80 | 26.60 | 28.20 | 21.60 | 29.56 |
| + LongCE | 84.02 | 71.50 | 3.55 | 5.60 | 36.00 | **34.80** | 34.60 | **32.60** | 29.40 | 33.48 |
| + CDT (ours) | **84.76** | **73.40** | **3.04** | **5.40** | **38.40** | 34.60 | **34.80** | 31.40 | **29.60** | **33.76** |
| Llama-3.1-8B-Base | 89.99 | 81.96 | 3.22 | **4.79** | 35.00 | 33.20 | 27.80 | 28.00 | 25.20 | 29.84 |
| + CE | 86.59 | 80.87 | 3.28 | 4.86 | **39.20** | 31.60 | 31.40 | 26.60 | 26.80 | 31.12 |
| + LongCE | 87.65 | 81.79 | 3.24 | 5.28 | 37.80 | 33.40 | **33.60** | **32.60** | 27.60 | 33.00 |
| + CDT (ours) | **90.36** | **82.23** | **2.10** | 5.19 | 38.80 | **36.60** | 33.20 | 29.40 | **28.20** | **33.24** |
| Llama-3.1-8B-Instruct | 92.49 | 85.98 | 4.05 | **5.52** | 46.60 | 49.60 | 42.40 | 38.80 | 37.00 | 42.88 |
| + SFT | 92.49 | 86.22 | 3.31 | 5.51 | 47.00 | 49.40 | **43.60** | 41.20 | 37.40 | 43.72 |
| + LOGO | 92.54 | 86.93 | 4.11 | 5.54 | 48.20 | 50.00 | 42.60 | 42.20 | 37.40 | 44.08 |
| + CDT (ours) | **93.08** | **88.01** | **2.36** | 5.64 | **51.40** | **51.20** | 41.60 | **44.00** | **38.60** | **45.36** |

Table 2: Evaluation results on long synthetic tasks (RULER), language modeling, and long-form reasoning (BABILong). For RULER, we report the average scores across 13 sampled sub-tasks. To calculate LongPPL, we apply the Llama3-8B-Base model as the evaluator. For BABILong, we report the model reasoning capability from short context (4K) to long context (64K). More evaluation results are shown in Appendix D.

trast, using standard CE or LongCE Loss leads to significant performance drops on some sub-tasks. For example, LongCE results in a nearly 4-point drop compared to the backbone model on the Few-shot subtask. **(3)** As for the LCM-Instruct models (the bottom group), we find that, due to its remarkable performance, existing post-training methods do not bring significant improvements. For instance, Llama-3.1-8B-SEALONG (48.90) achieves only around slight 0.3-point average improvement compared to Llama-3.1-8B-Instruct (49.61). However, our CDT achieves an average improvement of more than 2 points compared to the backbone across all tasks.

**Long Synthetic Task and Language Modeling** For the long synthetic task, we evaluate the model's performance under 32K and 64K context lengths. We choose 13 sub-tasks from the RULER benchmark and report the average results. For the language modeling task, we calculate both the LongPPL (Fang et al., 2024b) and PPL metrics on the GovReport dataset (Huang et al., 2021). More details are shown in Appendix C.2. As shown in Table 2, we can observe that our CDT method achieves the best model performance on the RULER benchmark on both 32K and 64K settings. For language modeling task, CDT exhibits the lowest LongPPL in language modeling tasks and demonstrates competitive results on the PPL metric. Notably, LongPPL can potentially reflect the model's ability to locate salient tokens in the long context, indicating the great potential of CDT.

**Short-context & Long-form Reasoning Tasks** We evaluate the model's long-form reasoning capabilities, as well as its short-context capability, on BABILong, a synthetic task that requires models to reason through multiple supporting facts hidden in contexts of varying lengths (from 4K to 64K). As shown in Table 2, our CDT achieves the highest overall score in each group. Besides, we observe that our CDT does not compromise the model's performance on short-context tasks. For instance, in the 4K and 8K lengths, CDT achieves either the best or comparable results compared to other methods and backbone models.

## 6 Ablation Study

In this section, we compare the accuracy of salient token detection of CDT with other detection methods in §6.1. Then, we show the impact of token embedding denoising on the training process in §6.2. Finally, we elaborate on the training efficiency of our CDT method in §6.3.

### 6.1 Comparison of Critical Token Detection

We compare three different detection methods, including LongPPL, attention-based detection, and our CDT, on our synthetic task (Figure 2). For attention-based and our CDT methods, we treat the tokens with the top-30 highest attention scores

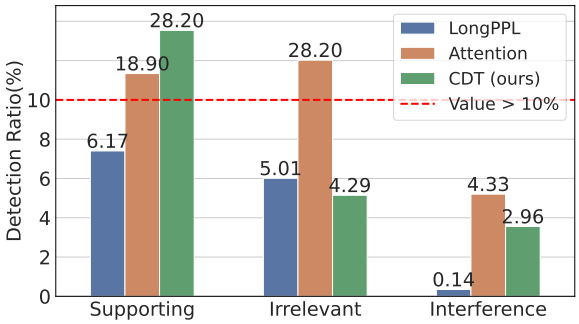



Figure 7: Comparison of critical token detection capability among different methods on our synthetic task.

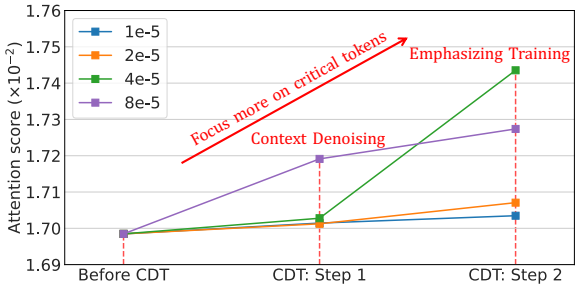



Figure 8: Impact of context denoising and comparison of the effect of learning rate on attention scores assigned to critical tokens in CDT.

and L2 normalized gradient of embedding as the detected tokens. As shown in Figure 7, we can observe that the attention-based method can detect a high proportion of supporting tokens and interference tokens, but it also detects a large number of irrelevant tokens. On the other hand, while LongPPL can effectively suppress the detection of irrelevant tokens, it struggles to locate supporting tokens. Our CDT method not only identifies the largest number of critical tokens but also effectively suppresses the detection of irrelevant tokens.

### 6.2 Impact of Context Denoising

In this section, we visualize the changes in attention scores allocated to critical tokens during the CDT training process. As shown in Figure 8, we observe that after the context denoising step, the attention scores on critical tokens have already increased significantly. Furthermore, after the Emphasizing Training stage, there is an additional improvement. Besides, we find that a larger learning rate leads to a more pronounced improvement, as it further enhances the denoising of the context.

### 6.3 Analysis of Training Efficiency

In addition to the performance improvement brought by CDT, we also demonstrate the efficiency

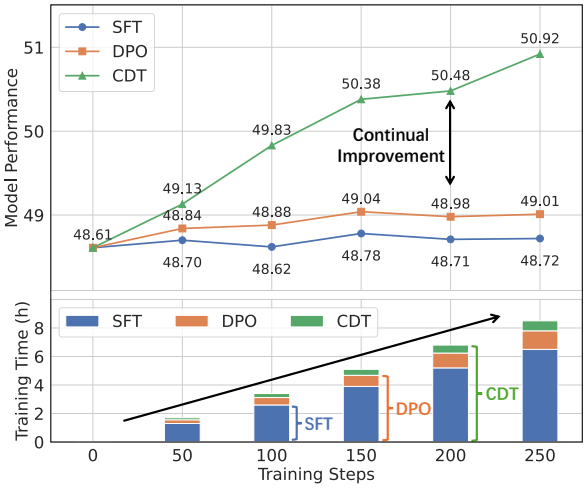



Figure 9: The performance improvement and training duration for every interval of 50 steps.

of our method. In CDT, the noise detection step introduces additional computation cost. We compare CDT with SFT (single Forward-Backward) and DPO (one batch contains pairwise samples) methods shown in Table 1. As shown in Figure 9, we observe that although CDT brings additional cost, i.e., approximately 0.5 hours in 8×A100 GPUs for every 50 steps compared with vanilla SFT, it consistently improves the model performance. With the same training steps, DPO only yields marginal improvements, while SFT even demonstrates a decline in performance. We provide the total training time in Appendix C. For each epoch, CDT takes 6.5 hours compared to SFT's 4 hours, which is acceptable given the performance improvements.

## 7 Conclusion

Existing work suggests that long-context models process long-context input in a *retrieval-then-generation* manner and the "retrieval-context" might be overwhelmed by excessive irrelevant tokens. This impairs the model's performance. In this paper, we conduct a fine-grained analysis of the context noise. We propose an effective critical token detection metric, IG score, and observe that models can better focus on critical tokens after restraining the context noise. Based on the above findings, we propose a Context Denoising Training (CDT) strategy, which can simultaneously strengthen model's attention on critical tokens and establish a stronger connection between salient tokens and the model prediction. Experiments on different models across 4 various task types demonstrate the superiority of our proposed method.

## Limitation

Due to the expectation maximization (EM) nature of CDT, it includes an additional context noise detection process, which introduces extra computational costs during the training phase. Although we have demonstrated in Section 6.3 that these additional costs are negligible compared to the performance gains, theoretically, *the noise detection cost will increase as the model size grows since it involves a complete forward-backward propagation process*. We leave this for future work, aiming to explore a simpler method for identifying the context noise or to develop more efficient model architectures. For example, designing specific network modules to handle noise, as proposed in Ye et al. (2024), could be a promising direction. Additionally, we observe that *the improvement brought by our method on complex reasoning tasks is not as significant as that on other tasks*, and we are yet to understand the relationship between this and the training data or the training objective function. In the future, we aim to further investigate the impact of context noise on the model's long-form reasoning abilities, as well as the relationship between the CDT strategy and the enhancement of the model's reasoning capabilities.

## References


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

Yushi Bai, Xin Lv, Jiajie Zhang, Yuze He, Ji Qi, Lei Hou, Jie Tang, Yuxiao Dong, and Juanzi Li. 2024a. 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. 2024b. 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, Jie Tang, and Juanzi Li. 2024c. Longbench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. *arXiv preprint arXiv:2412.15204*.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew Gormley. 2024. Unlimiformer: Long-range transformers with unlimited length input. *Advances in Neural Information Processing Systems*, 36.

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*.

Yukang Chen, Shaozuo Yu, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023c. Long alpaca: Long-context instruction-following models. https://github.com/dvlab-research/LongLoRA.

Zhi Chen, Qiguang Chen, Libo Qin, Qipeng Guo, Haijun Lv, Yicheng Zou, Wanxiang Che, Hang Yan, Kai Chen, and Dahua Lin. 2024. What are the essential factors in crafting effective long context multi-hop instruction datasets? insights and best practices. *arXiv preprint arXiv:2409.01893*.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. Longrope: Extending llm context window beyond 2 million tokens. *arXiv preprint arXiv:2402.13753*.

Chongzhou Fang, Ning Miao, Shaurya Srivastav, Jialin Liu, Ruoyu Zhang, Ruijie Fang, Ryan Tsang, Najmeh Nazari, Han Wang, Houman Homayoun, et al. 2024a. Large language models for code analysis: Do {LLMs} really do their job? In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 829–846.

Lizhe Fang, Yifei Wang, Zhaoyang Liu, Chenheng Zhang, Stefanie Jegelka, Jinyang Gao, Bolin Ding, and Yisen Wang. 2024b. What is wrong with perplexity for long-context language modeling? *arXiv preprint arXiv:2410.23771*.

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. 2024a. Data engineering for scaling language models to 128k context. In *Forty-first International Conference on Machine Learning*.

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. 2024b. Data engineering for scaling language models to 128k context. In *Forty-first International Conference on Machine Learning*.

Chaochen Gao, Xing Wu, Qi Fu, and Songlin Hu. 2024a. Quest: Query-centric data synthesis approach for long-context scaling of large language model. *arXiv preprint arXiv:2405.19846*.

Chaochen Gao, Xing Wu, Zijia Lin, Debing Zhang, and Songlin Hu. 2025. Nextlong: Toward effective long-context training without long documents. *arXiv preprint arXiv:2501.12766*.

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

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

Aryo Pradipta Gema, Chen Jin, Ahmed Abdulaal, Tom Diethe, Philip Teare, Beatrice Alex, Pasquale Minervini, and Amrutha Saseendran. 2024. Decore: decoding by contrasting retrieval heads to mitigate hallucinations. *arXiv preprint arXiv:2410.18860*.

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*.

Jian Hu, Xibin Wu, Zilin Zhu, Xianyu, Weixun Wang, Dehao Zhang, and Yu Cao. 2024. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. 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, Online. Association for Computational Linguistics.

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

Yannis Kopsinis and Stephen McLaughlin. 2009. Development of emd-based denoising methods inspired by wavelet thresholding. *IEEE Transactions on signal Processing*, 57(4):1351–1362.

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 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, Miami, Florida, USA. Association for Computational Linguistics.

Huayang Li, Pat Verga, Priyanka Sen, Bowen Yang, Vijay Viswanathan, Patrick Lewis, Taro Watanabe, and Yixuan Su. 2024a. Alr$\hat{2}$: A retrieve-then-reason framework for long-context question answering. *arXiv preprint arXiv:2410.03227*.

Siheng Li, Cheng Yang, Zesen Cheng, Lemao Liu, Mo Yu, Yujiu Yang, and Wai Lam. 2024b. Large language models can self-improve in long-context reasoning. *arXiv preprint arXiv:2411.08147*.

Hao Liu, Matei Zaharia, and Pieter Abbeel. 2023. Ring attention with blockwise transformers for near-infinite context. *arXiv preprint arXiv:2310.01889*.

Jiaheng Liu, Zhiqi Bai, Yuanxing Zhang, Chenchen Zhang, Yu Zhang, Ge Zhang, Jiakai Wang, Haoran Que, Yukang Chen, Wenbo Su, et al. 2024a. Eˆ 2-llm: Efficient and extreme length extension of large language models. *arXiv preprint arXiv:2401.06951*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024b. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 11:157–173.

Yi Lu, Jing Nathan Yan, Songlin Yang, Justin T Chiu, Siyu Ren, Fei Yuan, Wenting Zhao, Zhiyong Wu, and Alexander M Rush. 2024. A controlled study on long context extension and generalization in llms. *arXiv preprint arXiv:2409.12181*.

AI Meta. 2024. Introducing llama 3.1: Our most capable models to date. *Meta AI Blog*, 12.

MiniMax, Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, Enwei Jiao, Gengxin Li, Guojun Zhang, Haohai Sun, Houze Dong, Jiadai Zhu, Jiaqi Zhuang, Jiayuan Song, Jin Zhu, Jingtao Han, Jingyang Li, Junbin Xie, Junhao Xu, Junjie Yan, Kaishun Zhang, Kecheng Xiao, Kexi Kang, Le Han, Leyang Wang, Lianfei Yu, Liheng Feng, Lin Zheng, Linbo Chai, Long Xing, Meizhi Ju, Mingyuan Chi, Mozhi Zhang, Peikai Huang, Pengcheng Niu, Pengfei Li, Pengyu Zhao, Qi Yang, Qidi Xu, Qiexiang Wang, Qin Wang, Qiuhui Li, Ruitao Leng, Shengmin Shi, Shuqi Yu, Sichen Li, Songquan Zhu, Tao Huang, Tianrun Liang, Weigao Sun, Weixuan Sun, Weiyu Cheng, Wenkai Li, Xiangjun Song, Xiao Su, Xiaodong Han, Xinjie Zhang, Xinzhu Hou, Xu Min, Xun Zou, Xuyang Shen, Yan Gong, Yingjie Zhu, Yipeng Zhou, Yiran Zhong, Yongyi Hu, Yuanxiang Fan, Yue Yu, Yufeng Yang, Yuhao Li, Yunan Huang, Yunji Li, Yunpeng Huang, Yunzhi Xu, Yuxin Mao, Zehan Li, Zekang Li, Zewei Tao, Zewen Ying, Zhaoyang Cong, Zhen Qin, Zhenhua Fan, Zhihang Yu, Zhuo Jiang, and Zijia Wu. 2025. Minimax-01: Scaling foundation models with lightning attention. *Preprint*, arXiv:2501.08313.

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

Yifu Qiu, Varun Embar, Yizhe Zhang, Navdeep Jaitly, Shay B Cohen, and Benjamin Han. 2025. Eliciting in-context retrieval and reasoning for long-context large language models. *arXiv preprint arXiv:2501.08248*.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. 2019. Compressive transformers for long-range sequence modelling. *arXiv preprint*.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Zecheng Tang, Zechen Sun, Juntao Li, Qiaoming Zhu, and Min Zhang. 2024a. Logo–long context alignment via efficient preference optimization. *arXiv preprint arXiv:2410.18533*.

Zecheng Tang, Keyan Zhou, Juntao Li, Baibei Ji, Jianye Hou, and Min Zhang. 2024b. L-citeeval: Do long-context models truly leverage context for responding? *arXiv preprint arXiv:2410.02115*.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Label words are anchors: An information flow perspective for understanding in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9840–9855.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2024. Retrieval head mechanistically explains long-context factuality. *arXiv preprint arXiv:2404.15574*.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. 2024a. Duoattention: Efficient long-context llm inference with retrieval and streaming heads. *arXiv preprint arXiv:2410.10819*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024b. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*.

Zheyang Xiong, Vasilis Papageorgiou, Kangwook Lee, and Dimitris Papailiopoulos. 2024. From artificial needles to real haystacks: Improving retrieval capabilities in llms by finetuning on synthetic data. *arXiv preprint arXiv:2406.19292*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. 2024. Differential transformer. *arXiv preprint arXiv:2410.05258*.

Tan Yu, Anbang Xu, and Rama Akkiraju. 2024. In defense of rag in the era of long-context language models. *arXiv preprint arXiv:2409.01666*.

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*.

Jiajie Zhang, Zhongni Hou, Xin Lv, Shulin Cao, Zhenyu Hou, Yilin Niu, Lei Hou, Yuxiao Dong, Ling Feng, and Juanzi Li. 2024b. Longreward: Improving long-context large language models with ai feedback. *arXiv preprint arXiv:2410.21252*.

Yikai Zhang, Junlong Li, and Pengfei Liu. 2024c. Extending llms' context window with 100 samples. *arXiv preprint arXiv:2401.07004*.

Liang Zhao, Tianwen Wei, Liang Zeng, Cheng Cheng, Liu Yang, Peng Cheng, Lijie Wang, Chenxia Li, Xuejie Wu, Bo Zhu, et al. 2024a. Longskywork: A training recipe for efficiently extending context length in large language models. *arXiv preprint arXiv:2406.00605*.

Xinyu Zhao, Fangcong Yin, and Greg Durrett. 2024b. Understanding synthetic context extension via retrieval heads. *arXiv preprint arXiv:2410.22316*.

# A Preliminary Study Details

## A.1 Preliminary Task Construction

**Task Selection** We select 3-hop and 4-hop tasks based on qa3 tasks in the BABILong Benchmark to build our datasets, as these tasks generally pose significant challenges for LLMs. However, it is worth noting that the original BABILong qa4 samples do not truly require 4-hop reasoning to produce correct outputs. For example, a sample from this subset with 0k context is shown in Figure 10. In this case, the task only requires attention to a single fact, "The bedroom is west of the bathroom" to answer the question, while the first sentence serves as an interference fact. Even in terms of keywords, the model only needs to focus on three keywords: "bathroom", "west", and "bedroom" from the second sentence. Thus, we design our 4-hop dataset based on the BABILong qa3 source data, with one sample shown in Figure 11. By carefully arranging the order of facts and reducing the conditions of questions in the long context, we ensure that the model is required to search for all four supporting facts in sequence to produce the correct output.

**Controlled Evaluation Data Synthesis** We use the 4-hop task with non-zero context as an example here. As shown in Table 3, all variables used for building data include the facts sample, the facts permutation, and the context length. Firstly, we select source samples from the BABILong official file "qa3_three-supporting-facts" as our base data. Then, we modify the original BABILong qa3 supporting facts following the pattern shown in Figure 12. Afterward, we add interference to these four original facts while maintaining the relative order of the supporting facts. The process begins by selecting a noise context of the appropriate length and inserting the facts into it. Specifically, we divide the noise context into 10 equal-length chunks, leaving 10 candidate positions for the insertion of the 4 supporting facts (excluding the tail). Next, we randomly select five permutations from the full set of $C_{10}^4$ candidate position permutations. After injecting noise, we randomly insert interference facts, i.e., facts that are similar to the supporting facts but irrelevant, among all sentences. We ensure that at least one interference fact is placed after the last supporting fact to test the model's robustness. To ensure the correctness of the samples, we make sure that the objects appearing in the interference facts do not overlap with those in the supporting facts. Additionally, we ensure that the number of interference facts is between one and two times the number of supporting facts to avoid making the samples either too easy or too difficult. Finally, for all samples with the same context length, we use the same noise context to maintain consistency. In the end, we randomly insert a few emojis into the constructed context to test the sensitivity of the model to low-frequency tokens. For the 3-hop task, we directly use the original qa3 task format from BABILong as the base, and the subsequent processing follows a similar approach to the one described above for the 4-hop task.

| Hops | Samples | Permute | Lengths |
|---|---|---|---|
| 2 | 100 | 5 | 8K |
| 3/4 | R. | R. | 0k - 64k |

Table 3: Variable settings, where R. denotes random.

**One BABILong qa4 sample with 0k context**

Input :

```
The bedroom is west of the office.
The bathroom is west of the bedroom.
```

***Question:***
What is west of the office?

***Supporting Facts:***
The bedroom is west of the office.

***Ground truth:***
bedroom

Figure 10: A BABILong qa4 sample with 0k context

# B Derivation of Relation between Information Flow and Embedding Gradients

In transformer-based models, the Information Flow in attention is essentially the product of the attention distribution and its corresponding gradient. Therefore, we can transform the derivation into **constructing the gradient relationship between the attention score distribution ($A$) and the embedding ($E(X)$)**. This can be established via the chain rule and implemented through the specific computation steps of the attention mechanism. Notably, in the following derivation, for simplicity, we omit the activation layers in the model. Additionally, considering that transformer-based

**One of our 4-hop samples with 0k context**

Input :

```
Mary journeyed to the office.
Mike went to the office.
Mary got the apple.
Daniel picked up the football.
Daniel went back to the bedroom.
Mary journeyed to the bathroom.
Mary dropped the apple.
Jonh went to the bathroom.
```

***Question:***
Where was the apple's location prior to the place where the apple was discarded, left or dropped?

***Supporting Facts:***
Mary journeyed to the office.
Mary got the apple.
Mary journeyed to the bathroom.
Mary dropped the apple.

***Ground truth:***
office

Figure 11: One of our 4-hop samples with 0k context

**The pattern of our 4-hop sample**

```
Supporting fact1: {x} {m} the {y1}
Supporting fact2: {x} {p} the {o}
Supporting fact3: {x} {m} the {y2}
Supporting fact4: {x} {d} the {o}
```

***Question:***
Where was the **{o}**'s location prior to the place where the **{o}** was discarded, left or dropped?

***Ground truth:***
**{y1}**

***Explanation:***
**{x}** : a character name, selected from {Mary, Daniel, Mike, ...}
**{m}** : a predicate indicating movement, selected from {went to, journeyed to, travelled to, ...}
**{y1}, {y2}** : two different locations, selected from {office, bedroom, bathroom, ...}
**{p}** : a predicate indicating picking up, selected from {picked up, took, grabbed, ...}
**{d}** : a predicae indicating dropping, selected from {dropped, put down, discarded, ...}
**{o}** : an object name, selected from {apple, football, milk, ...}

Figure 12: The pattern of our 4-hop sample

models are composed of multiple identical network blocks stacked together, one can easily extend the conclusions from a single layer to multiple layers. Therefore, we focus on proving the case with **one embedding layer and one attention module**.

Given the basic definition of the attention mechanism, we have:

$$\begin{cases} Q = E(X)W_Q, & A = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right), \\ K = E(X)W_K, & O = A \cdot V, \\ V = E(X)W_V, & \end{cases}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d\times d}$ are the model parameters, $O$ is the attention output, $E(X) \in \mathbb{R}^{n\times d}$ is the input embedding matrix, $n$ and $d$ are sequence length and model dimension, respectively.

Let the loss function be $L$. By the chain rule, the gradient of the loss with respect to $E(X)$ is:

$$\frac{\partial L}{\partial E(X)} = \frac{\partial L}{\partial O}\frac{\partial O}{\partial E(X)} = \frac{\partial L}{\partial A}\frac{\partial A}{\partial E(X)} + \frac{\partial L}{\partial V}\frac{\partial V}{\partial E(X)}. \quad (4)$$

Since we have $\frac{\partial V}{\partial E(X)} = W_V^T$ and $\frac{\partial O}{\partial V} = A$, the gradient relationship between $A$ and $E(X)$ is:

$$\frac{\partial L}{\partial E(X)} \propto \frac{\partial L}{\partial A}\frac{\partial A}{\partial E(X)} \quad (5)$$

To eliminate the influence of the $\text{Softmax}(\cdot)$ function, we can further decompose equation 5 into:

$$\begin{cases} S = \dfrac{QK^T}{\sqrt{d}}, \\ \dfrac{\partial L}{\partial E(X)} \approx \dfrac{\partial L}{\partial A} \cdot \left(\dfrac{\partial A}{\partial S} \cdot \dfrac{\partial S}{\partial E(X)}\right), \end{cases} \quad (6)$$

where $\frac{\partial A}{\partial S}$ is the Jacobian of $\text{Softmax}(\cdot)$ function, with elements $A_{ij}\left(\delta_{ik} - A_{ik}\right)$[4].

For each element $S_{ij} = \frac{Q_i K_j^T}{\sqrt{d}} \in S$, the gradient with respect to $E(X)$ can be written as:

$$\frac{\partial S_{ij}}{\partial E(X)} = \frac{\partial\left(\frac{(E(X)_i W_Q)(E(X)_j W_K)^T}{\sqrt{d}}\right)}{\partial E(X)} = \frac{1}{\sqrt{d}}\left(W_Q^T \cdot K_j \cdot \delta_{ik} + W_K^T \cdot Q_i \cdot \delta_{jk}\right). \quad (7)$$

[4] $\delta_{ik}$ is the Kronecker delta function. If $i$ equals to $k$, $\delta_{ik} = 1$, else $\delta_{ik} = 0$. We can also rewrite this equation into $A_{ij}\left(1 - A_{ij}\right)$.

Based on equation 6 and equation 7, we can summary that:

$$\frac{\partial L}{\partial E(X)_i} \propto \underbrace{\frac{\partial L}{\partial A_{ij}}}_{\texttt{Sensitivity of } L \texttt{ to } A} \times \underbrace{A_{ij}(1-A_{ij})}_{\texttt{Derivation from Softmax}} \times \underbrace{\frac{\partial S_{ij}}{\partial E(X)}}_{\texttt{Linear Transformation}} . \tag{8}$$

Based on equation 8, we can derive that when $A_{ij}$ increases, indicating higher attention between token $i$ and token $j$, the sensitivity of $L$ to $A$ ($\frac{\partial L}{\partial A_{ij}}$) also increases. This results in larger derivatives on the embeddings. Additionally, if $A_{ij}$ becomes excessively large, approaching 1, the value of $A_{ij}(1-A_{ij})$ might tend toward 0. However, this is often not an issue in long-context scenarios, as the attention scores are unlikely to approach values near 0.5 due to the long context. Even if they exceed 0.5 (possibly for some special tokens), the increase in the first term ($\frac{\partial L}{\partial A_{ij}}$) helps mitigate this effect.

# C Implementation Details

## C.1 Training Details

For all experiments, we utilize the open-source training framework OpenRLHF[5] (Hu et al., 2024), Ring-flash-attention[6] (Liu et al., 2023) and DeepSpeed (Rajbhandari et al., 2020). For LongCE training (Fang et al., 2024b), we set the sliding context window size as 8192 and employ the recommended hyper-parameters in the official code [7].

**Context Window Scaling** To scale the context window size of the Llama-3-8B-base model from 8K to 64K (8×), we adjust the RoPE base from 500,000 to 20,000,000 and directly train the model. We provide training configurations in Table 4.

**Language Modeling Post-training and Long-context SFT** The language modeling post-training and long-context SFT are directly applied to the Llama3.1-8B-base and Llama3.1-8B-Instruct, respectively, which already have 128K context window size. We provide the training configurations in Table 5 and Table 6 respectively.

[5] https://github.com/OpenRLHF/OpenRLHF.git

[6] https://github.com/zhuzilin/ring-flash-attention.git

[7] https://github.com/PKU-ML/LongPPL.git

| Context Window Scaling Training Setting | |
|---|---|
| Backbone | Llama-3-8B-base |
| Training Objective | Language modeling |
| RoPE base | 20,000,000 |
| Context window size | 8K → 64K |
| Data seq-length | 64,000 |
| Deepspeed | Zero2 |
| Global batch size | 64 |
| Epoch | 2 |
| Training Steps | 160 |
| Ring-attention size | 4 |
| Learning-rate | 1e-5 |
| LR-scheduler | cosine_with_min_lr |
| Optimizer | Adam ($\beta_1 = 0.9, \beta_2 = 0.95$) |
| GPUs | A100 (80GB) × 8 |
| Training time | ≈8h / epoch |
| Training data | PG19 (Rae et al., 2019) |
| Total consumed tokens | 0.65B |

Table 4: Configuration of context window scaling.

## C.2 Evaluation Details

We conduct long-context evaluation mainly based on the open-source evaluation framework[8].

**LongBench-E** LongBench-E is a variant of LongBench (Bai et al., 2024b) designed specifically for long-context real-world tasks. We chose LongBench-E because it shares the same test dataset distribution as LongBench while covering a wider range of context lengths. For the Llama3-8B-base model, we truncate the input to 8K tokens, whereas for other models, we truncate the input to 32K tokens.

**Language Modeling** For the language modeling task, we calculate both LongPPL and PPL metrics on the GovReport dataset (Huang et al., 2021), which consists of long sequences from government reports. We sample 50 documents from GovReport, each with a context length of up to 32K tokens.

**RULER** RULER (Hsieh et al., 2024) is a comprehensive synthetic dataset that includes 6 different testing categories to evaluate a model's long-context understanding capabilities. We utilize all test categories, with each category containing 50 test samples covering lengths of 32K and 64K. We post the testing configuration of RULER in Table 8.

**Long-form Reasoning** We evaluate the long-form reasoning capability of models on selected tasks from BABILong (Kuratov et al., 2024).

[8] https://github.com/ZetangForward/Long-context-Eval.git

| Language Modeling Post-training Setting | |
|---|---|
| Backbone | Llama-3.1-8B-base |
| Training Objective | Language modeling |
| RoPE base | 500,000 |
| Context window size | 128K |
| Data seq-length | 64,000 |
| Deepspeed | Zero2 |
| Epoch | 2 |
| Global batch size | 32 |
| Training Steps | 320 |
| Ring-attention size | 4 |
| Learning-rate | 5e-6 |
| LR-scheduler | cosine_with_min_lr |
| Optimizer | Adam ($\beta_1 = 0.9, \beta_2 = 0.95$) |
| GPUs | A100 (80GB) $\times$ 8 |
| Training time | $\approx$8.5h / epoch |
| Training data | PG19 (Rae et al., 2019) |
| Total consumed tokens | 0.65B |

Table 5: Configuration of language modeling.

| Long-context Alignment Training Setting | |
|---|---|
| Backbone | Llama-3.1-8B-Instruct |
| Training Objective | Supervised fine-tuning |
| RoPE base | 500,000 |
| Context window size | 128K |
| Data seq-length | 4,000$\sim$128,000 |
| Deepspeed | Zero2 |
| Global batch size | 32 |
| Epoch | 2 |
| Training Steps | 250 |
| Ring-attention size | 4 |
| Learning-rate | 5e-6 |
| LR-scheduler | cosine_with_min_lr |
| Optimizer | Adam ($\beta_1 = 0.9, \beta_2 = 0.95$) |
| GPUs | A100 (80GB) $\times$ 8 |
| Training time | $\approx$6.5h / epoch |
| Training data | LongMIT (Chen et al., 2024), LongAlpaca (Chen et al., 2023c) |
| Total consumed tokens | 0.53B |

Table 6: Configuration of long-context SFT training.

Specifically, we select tasks that involve multiple supporting facts, as well as QA1, as the testing dataset. The BABILong testing configurations are shown in Table 10.

# D More Evaluation Results

## D.1 Generalizing CDT to Longer Context Length

We generalize the long-context evaluation to 128K context size on RULER (128K) and BABILong (128K) benchmarks. As shown in Table 9, we can find that our CDT method still outperforms other methods and strong LCMs.

## D.2 Generalizing CDT to More Models

We apply our CDT method to more LLMs, including Qwen2.5-7B-Instruct (Yang et al., 2024) and Mistral-V0.3-Instruct (Jiang et al., 2023). We evaluation the model performance on real-world long-context tasks, long synthetic tasks, and long-form reasoning tasks. We report the model performance in Table 7, where we can observe that our CDT can significantly improve the model performance on different models. For instance, the Mistral-V0.3-Instruct model obtains more than 30 points on the long-form reasoning task.

# E Error Analysis

In this section, we analyze the error pattern of partial model predictions on real-world long-context tasks. As shown in Table 11, we use colored text to highlight the correct and incorrect parts of the model's predictions.

| Models | LongBench-E | | | | | | | BABILong |
|---|---|---|---|---|---|---|---|---|
| | Type | S-Doc QA | M-Doc QA | Summ | Few-shot | Code | Avg. | Avg. |
| Qwen2.5-7B-Instruct | - | 44.54 | 46.29 | 28.15 | 56.03 | 16.52 | 38.30 | 43.32 |
| + CDT | SFT | **44.93** | **47.29** | **28.65** | **57.33** | **19.18** | **39.48** | **47.56** |
| Mistral-V0.3-Instruct | - | 44.89 | 40.76 | 20.52 | 67.11 | 47.04 | 44.06 | 22.36 |
| + CDT | SFT | **45.01** | **41.79** | **26.08** | **67.75** | **57.27** | **47.58** | **53.84** |

Table 7: Evaluation results of two more LLMs on real-world long-context tasks and long-form reasoning tasks.

| Evaluation Configuration of RULER | |
|---|---|
| Question Answering | qa_1, qa_2 |
| Single NIAH | niah_single_1,<br>niah_single_2,<br>niah_single_3 |
| Multi-keys NIAH | niah_multikey_1,<br>niah_multikey_2,<br>niah_multikey_3 |
| Multi-values NIAH | niah_multiquery |
| Multi-queries NIAH | niah_multivalue |
| Others | common words extraction (CWE),<br>frequent words extraction (FWE),<br>variable tracking (VT) |
| Length | 32K, 64K |
| Num samples/task | 50 |

Table 8: Testing configuration of RULER

| Models | RULER | BABILong |
|---|---|---|
| ProLong-512K-Instruct | 72.13 | 21.00 |
| NExtLong-512K-Instruct | 76.89 | 22.00 |
| Llama-3.1-8B-SEALONG | 86.63 | 31.00 |
| Llama-3.1-8B-Base | 81.96 | 24.40 |
| + CE | 80.87 | 19.40 |
| + LongCE | 81.79 | 24.60 |
| + CDT (ours) | **82.23** | **28.20** |
| Llama-3.1-8B-Instruct | 85.98 | 29.60 |
| + SFT | 86.22 | 30.40 |
| + LOGO | 86.93 | 31.60 |
| + CDT (ours) | **88.01** | **38.60** |

Table 9: Model performance on tasks with 128K context length.

| Task | Num | Supporting Fact | Interference Fact |
|---|---|---|---|
| QA1 | 100 | 1 | 1∼9 |
| QA2 | 100 | 2 | 1∼66 |
| QA3 | 100 | 3 | 1∼317 |
| QA7 | 100 | 1∼10 | 1∼42 |
| QA8 | 100 | 1∼8 | 1∼42 |

Table 10: Testing configuration of BABILong

| Code Completion Task | |
|---|---|
| **Question**: Completing the code based on the context: [···]\n self.interface = self.interface.replace(' ', '').lower() \n self.param_type = None \n self.if_type = None \n # state \n self.results = dict() \n [Completion] self.result = dict() \n def check_params(self): \n """Check all input params""" \n if not self.interface: \n self.module.fail_json(msg='Error: Interface name cannot be empty.')[···] | **Llama3.1-8B-Instruct**: for (int i = 0; i < s.length(); i++) {if (!Character.isDigit(s.charAt(i)) && s.charAt(i) != ',') }<br>**Llama3.1-8B-CD (ours)**: if self.interface == 'all':\n self.param_type = INTERFACE_ALL\n<br>**GPT4o**: if self.interface.lower() == 'all':\n self.param_type = INTERFACE_ALL\n elif '/' in self.interface or self.interface.isdigit():\n self.param_type = INTERFACE_FULL_NAME\n else:\n self.if_type = get_interface_type(self.interface)\n<br>**Golden Answer**: if self.interface == 'all':\n |
| **Long-context Multi-hop Reasoning Task** | |
| **Question**: What did M. King Hubbert's employer announce it was in the process of doing in April 2010? [···] In April 2010 Shell announced that the corporation is in process of trying to find a potential buyer for all of its operations in Finland and is doing similar market research concerning Swedish operations. [···] | **Llama3.1-8B-Instruct**: Shell announced the sale of some of its assets, including its liquefied petroleum gas (LPG) business, to meet the cost of a planned $28<br>**Llama3.1-8B-CD (ours)**: trying to find a potential buyer for all of its operations in Finland and is doing similar market research concerning Swedish operations.<br>**GPT4o**: MV Miralda's owner announced it was in the process of trying to find a potential buyer for all of its operations in Finland and in doing similar market research concerning.<br>**Golden Answer**: trying to find a potential buyer for all of its operations in Finland |
| **Single-QA Long-context Task** | |
| **Question**: What metrics are used in evaluation? [···] Figure FIGREF22 shows the results of these experiments wherein we are able to maintain the **precision** on the subset of the training set used for testing while reducing its **recall** and thus generalising the model. The fact that the model is generalizing on a dissimilar dataset is confirmed by the increase in the development set. **F1 score**. [···] | **Llama3.1-8B-Instruct**:<br>Macro-averaged F-score.<br>**Llama3.1-8B-CD (ours)**:<br>macro-averaged F-score, precision, recall, F1 score.<br>**GPT4o**:<br>Macro-averaged F-score.<br>**Golden Answer**:<br>precision, recall , F1 score |

Table 11: Generation samples of Llama3.1-8B-Instruct, Llama3.1-8B-Instruct-CD (ours), and GPT4o on real-world long-context tasks. We can observe that the Llama3.1-8B-Instruct model fails to generate the correct answers across the three long-context tasks. In contrast, both GPT4o and our method successfully cover the correct answers, although they may include some irrelevant content.