Beyond Prefix Answering: Parallel Infilling Reading for Long Contexts with Diffusion Language Models

Anonymous ACL submission

Abstract

Diffusion Language Models (DLMs) have emerged as a promising alternative to autoregressive language models, showing strong non-autoregressive generation abilities. Although the masked denoising objective trains DLMs to recover tokens at arbitrary positions, current long-context usage still mostly follows a prefix-based paradigm: the full context is given as a prefix, and the answer is generated only at the end, leaving the model's infilling ability underutilized. To investigate whether DLMs can read long contexts by infilling, we introduce **P**arallel **I**nfilling **R**eading (**PIR**), a training-free method that inserts extraction slots into long contexts to generate intermediate reading states. Our empirical study shows that naive PIR improves over prefix-based DLM inference, but suffers from slot repetition, where multiple slots generate redundant evidence. To mitigate this issue, we propose **Slot-Isolated PIR**, which blocks direct slot-to-slot attention, and **PIR-SFT**, which uses fine-grained evidence supervision to teach DLMs effective evidence extraction while reading. Experiments on long-context question answering benchmarks show that our methods improve answer accuracy and generation efficiency, with increasingly larger gains as the context length grows.

1 Introduction

Recently, Diffusion Language Models (DLMs) have attracted increasing attention as an alternative paradigm to autoregressive (AR) language models (Li et al., 2025). Unlike AR models that generate text through left-to-right prediction, DLMs generate text via iterative masked denoising with bidirectional attention. This mechanism naturally supports parallel decoding and arbitrary-order generation, and has shown promise in reasoning, code generation, and structured generation tasks (Nie et al., 2025; Ye et al., 2025; Lee et al., 2025; Wu et al., 2026). However, existing DLM studies mainly focus on short-context settings. How to effectively utilize long-context information with DLMs remains an important open problem.

Existing work on long-context DLMs mainly follows the context extension route developed for AR models (Chen et al., 2023; Peng et al., 2024; Liu et al., 2024; Dong et al., 2025; Liu et al., 2026; He et al., 2025). Although these methods extend the accessible context length, they still largely adopt a prefix-based prompting format during instruction tuning and downstream inference: the full context is provided as a prefix, and the answer is generated only at the end of the sequence. This usage is not fully aligned with the masked denoising objective of DLMs. Since DLMs are trained to recover masked tokens at arbitrary positions, they are naturally suitable for infilling, where the model generates missing content conditioned on surrounding contexts (Wu et al., 2026). However, in existing long-context usage, this infilling ability is rarely used for the reading process itself. For long-context tasks, where the model needs to locate and integrate evidence distributed across a long input, a more natural interface is to let DLMs generate intermediate reading states inside the context before producing the final answer. This motivates our empirical question: *Can vanilla DLMs read by infilling?*

To answer this question, we design a training-free inference method, **Parallel Infilling Reading (PIR)** which decomposes long-context question answering (QA) into *Parallel Evidence Extraction* and *Answer Reduction*: PIR first fills extraction slots after different context chunks in parallel to obtain question-relevant reading states, and then aggregates these states to produce the final answer. Our empirical study shows that even without additional training, naive PIR outperforms prefix-based DLM inference, suggesting that the infilling ability of DLMs can indeed be used for long-context reading. However, further analysis reveals that naive PIR still suffers from clear **slot repetition**: multi-

ple extraction slots tend to generate similar or even repeated candidate evidence, leading to degraded extraction quality. Attention probing further shows that this issue is associated with cross-slot attention interference among different slots.

Based on these findings, we improve PIR from both inference-time and training-time perspectives. First, we propose training-free **Slot-Isolated PIR**, which blocks direct slot-to-slot attention among different extraction slots during the Parallel Evidence Extraction stage, thereby mitigating cross-slot copying and redundant extraction. Further, we propose **PIR-SFT**, which constructs fine-grained evidence supervision and trains the model to extract useful evidence while reading through single-active-slot training. Compared with Slot-Isolated PIR, which relies on a custom attention mask, PIR-SFT internalizes the evidence extraction ability into the model, allowing inference to follow the standard PIR decoding procedure while improving extraction quality and inference efficiency.

Extensive experiments on long-context question answering benchmarks show that Slot-Isolated PIR improves DLMs' long-context reading ability, and PIR-SFT further enhances question answering performance while maintaining higher inference efficiency. Further analysis shows that PIR-SFT substantially reduces slot repetition and improves the quality of extracted evidence. These results validate infilling as an effective reading-time evidence extraction interface for DLMs.

2 Background

Compared with AR models, DLMs follow distinct training and inference paradigms (Nie et al., 2025). In this section, we briefly formalize their pre-training, supervised fine-tuning, and inference.

Pre-training. During pre-training, DLMs randomly mask a subset of tokens from the original text and train the model to recover the masked positions. Specifically, given the original sequence x_0, a noised sequence x_t at diffusion timestep t, and the masked position set $\mathcal{M}_t$, the pre-training objective can be simplified as:

$$\mathcal{L}_{\text{pre}} = -\mathbb{E}_{x_0,t,x_t} \sum_{i \in \mathcal{M}_t} \log p_\theta(x_{0,i} \mid x_t). \quad (1)$$

This objective enables DLMs to recover missing content by leveraging bidirectional attention over the partially observed sequence.

Supervised Fine-tuning. During supervised fine-tuning, existing DLMs typically adopt an instruction-response format. For a prompt p and a response r, the prompt remains visible while the response region is masked and used as the supervision target:

$$\mathcal{L}_{\text{resp}} = -\mathbb{E}_{(p,r),t} \sum_{i \in \mathcal{R}_t} \log p_\theta(r_i \mid p, \tilde{r}^{(t)}), \quad (2)$$

where $\tilde{r}^{(t)}$ denotes the partially masked response at noise level t, and $\mathcal{R}_t$ denotes the masked positions within this response region. Different from pre-training, supervised fine-tuning typically concentrates the mask prediction objective on the suffix response region.

Inference. Unlike AR models that generate text token by token in a left-to-right manner, DLMs start from a partially masked sequence at inference time and progressively recover masked tokens through iterative denoising. Given a sequence $x = (x_1, \ldots, x_n)$, let O denote the observed positions and M denote the masked positions. The initial inference sequence can be written as:

$$\tilde{x}_i^{(T)} = \begin{cases} x_i, & i \in O, \\ \texttt{[MASK]}, & i \in M. \end{cases} \quad (3)$$

The conditional generation objective of DLMs can be formulated as:

$$p_\theta(x_M \mid x_O), \quad (4)$$

where M is not restricted to a contiguous response region at the end of the sequence, but can correspond to masked spans at arbitrary positions. At denoising step t, let $\tilde{x}^{(t)}$ denote the current partially masked sequence and $\mathcal{M}^{(t)}$ denote the set of positions that remain masked. The model predicts tokens for all masked positions in parallel:

$$p_\theta(x_i \mid \tilde{x}^{(t)}), \;\; i \in \mathcal{M}^{(t)}. \quad (5)$$

The inference algorithm then selects a subset of masked positions to update according to confidence scores or a predefined schedule, and proceeds to the next denoising step. After multiple iterations, all masked positions are gradually filled. Since $\mathcal{M}^{(t)}$ can contain multiple disjoint positions, DLMs are able to handle multiple masked spans within the same sequence.

3 Empirical Study

To study whether DLMs can exploit infilling for long-context reading, we propose **Parallel Infilling Reading (PIR)**, an inference-time method that

enables DLMs to extract evidence from different parts of a long context in parallel. We then evaluate PIR on long-context QA benchmarks and analyze its main advantages and limitations.

3.1 Parallel Infilling Reading

PIR decomposes long-context QA into *Parallel Evidence Extraction* and *Answer Reduction*: the former fills extraction slots after context chunks to obtain reading states, while the latter aggregates these states to produce the final answer.

Parallel Evidence Extraction. In long-context QA, given a question Q and a long context C, the supporting evidence for the answer is often distributed across different positions. To extract such evidence, we first evenly partition the context into multiple chunks based on either the number of documents or a fixed token budget:

$$C = (c_1, c_2, \ldots, c_K), \tag{6}$$

where c_i denotes the i-th context chunk. We then insert an extraction slot after each chunk and concatenate the question, chunks, and slots into the PIR input:

$$\tilde{x}_{\text{PIR}} = [c_1; s_1; c_2; s_2; \ldots; c_K; s_K; Q], \tag{7}$$

where s_i denotes the i-th extraction slot, which consists of a short extraction prompt followed by m consecutive `[MASK]` tokens. The prompt instructs the model to extract question-relevant evidence from the corresponding chunk. During inference, PIR keeps the original denoising strategy of the DLM within each extraction slot, while decoding all slots in parallel. Specifically, all chunks and extraction slots remain in the same denoising canvas, so each slot can access the global raw context during generation. Each slot updates its own `[MASK]` tokens independently using the original DLM denoising rule, enabling PIR to generate multiple chunk-level reading states in one extraction pass.

Answer Reduction. We denote the generated content of the i-th slot as z_i, which serves as a question-relevant intermediate reading state. After extraction, we construct a fixed reduced input by concatenating the question and all generated slots:

$$\tilde{x}_{\text{red}} = [z_1; z_2; \ldots; z_K; Q; s_{\text{ans}}], \tag{8}$$

where $s_{\text{ans}} = [\texttt{[MASK]}]^{m_a}$ denotes the answer slot. The DLM then fills s_{ans} to integrate the extracted evidence and produce the final answer. Full extraction and reduction templates are provided in Appendix A.1 and Appendix A.2.

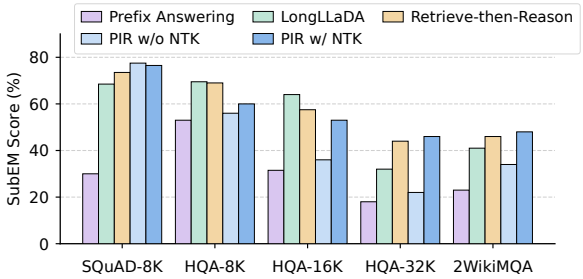

Figure 1: Empirical study of naive PIR on long-context QA benchmarks.

3.2 Can DLMs Read by Infilling?

We evaluate PIR with and without NTK-RoPE (Xiong et al., 2024) on LLaDA-8B-Instruct (Nie et al., 2025) across five long-context QA benchmarks, comparing it with three training-free baselines: Prefix Answering, LongLLaDA (Liu et al., 2026), and Retrieve-then-Reason (Li et al., 2024). Detailed benchmark and baseline settings are provided in Appendix B. As shown in Figure 1, PIR outperforms LLaDA across all evaluated benchmarks. After further incorporating NTK-RoPE, PIR even surpasses LongLLaDA and Retrieve-then-Reason on HotpotQA-32K and 2WikiMQA-16K, presenting stronger performance in longer-context settings. However, PIR w/ NTK-RoPE still lags behind some baselines on HotpotQA-8K and HotpotQA-16K. These results suggest that **although naive PIR verifies the feasibility of multi-slot infilling reading on long-context QA tasks, it has not yet fully unleashed its potential**.

3.3 Naive PIR Suffers from Slot Repetition

Slot Repetition. Although naive PIR shows promising potential, we observe a major extraction failure during evaluation: **slot repetition**. Specifically, multiple extraction slots often generate highly similar or even repeated candidate evidence, and in some cases repetition also occurs within a single slot. Since different slots are expected to extract relevant evidence from their corresponding chunks, such repetition wastes the multi-slot extraction capacity of PIR. Detailed case studies are provided in Appendix C.

Attention Probing. To better understand slot rep-

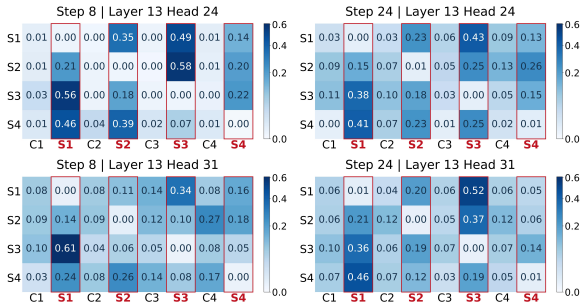

Figure 2: Attention probing of naive PIR during denoising. Rows are extraction slots as queries, and columns are context chunks or extraction slots as keys. Red boxes highlight slot-to-slot attention.

etition, we conduct attention probing during the diffusion decoding process. We group the input into chunk blocks $\{c_i\}$ and slot blocks $\{s_i\}$, and compute the block-level attention mass from each query slot to different key blocks. As shown in Figure 2, extraction slots in naive PIR often assign substantial attention to other slot blocks, rather than only to their corresponding chunks or their own slot. Such attention patterns appear across different denoising steps and heads, suggesting that slots can directly influence each other during generation and gradually converge to similar extraction results. We attribute this issue to the mismatch between prefix-based DLM SFT and multi-slot PIR inference: standard instruction tuning supervises a suffix response region, but provides little signal for independent context-side extraction slots.

4 Method

Based on the empirical analysis in Section 3, we find that PIR can elicit the long-context understanding ability of DLMs by inserting extraction slots inside the context. However, naive PIR is still mainly limited by **slot repetition**: multiple extraction slots tend to generate similar or even repeated candidate evidence, wasting the multi-slot extraction capacity. To further improve PIR, we enhance it from both inference-time and training-time perspectives. First, we propose training-free **Slot-Isolated PIR**, which blocks direct slot-to-slot attention among different extraction slots. Second, we introduce **PIR-SFT**, which trains the model with fine-grained evidence supervision to extract useful evidence while reading. Overall illustration is shown in Figure 3.

4.1 Slot-Isolated PIR

As analyzed in Section 3, naive PIR suffers from severe slot repetition during parallel denoising, which is likely caused by cross-slot attention that makes different extraction slots influence each other and converge to similar evidence. To mitigate this issue while allowing each extraction slot to access global context information, we propose a training-free method, *i.e.,* Slot-Isolated PIR. It follows the same PIR template and decoding procedure as Section 3.1. During the *Parallel Evidence Extraction* stage, we replace the default attention mask with a token-level slot isolation mask to block direct attention between different extraction slots, while keeping the answer reduction stage unchanged. Specifically, for the PIR input $\tilde{x}_{\text{PIR}}$, let $\mathcal{S}_i$ denote the token-index set of the i-th extraction slot s_i. We define the slot-isolated attention mask as:

$$M^{\text{iso}}_{q,k} = \begin{cases} -\infty, & q \in \mathcal{S}_i,\ k \in \mathcal{S}_j,\ i \neq j, \\ 0, & \text{otherwise}, \end{cases} \tag{9}$$

where q and k denote the query-token index and key-token index in the attention matrix, respectively. Here, 0 indicates normal attention, while $-\infty$ masks the corresponding attention score before softmax. This mask allows each slot to attend to the question, all context chunks, and its own generated tokens, while preventing direct cross-slot interactions. During inference, Slot-Isolated PIR follows the same parallel denoising procedure as naive PIR in Section 3.1, with the slot isolation mask applied throughout extraction. By reducing direct attention among slots, this method encourages each slot to generate a reading state closely aligned with its corresponding chunk, improving extraction quality.

4.2 PIR-SFT

Although Slot-Isolated PIR reduces cross-slot interference, it still relies on the zero-shot infilling ability of the original DLM and requires a custom attention mask, which slows down inference. To address these issues, we propose PIR-SFT. We first construct fine-grained evidence supervision for each extraction slot, and then use it to train the model to extract useful evidence while reading. Since PIR-SFT internalizes the extraction ability into the model, it avoids custom attention masks and keeps inference efficient.

Fine-Grained Evidence Supervision. To obtain precise training signals, we construct fine-grained

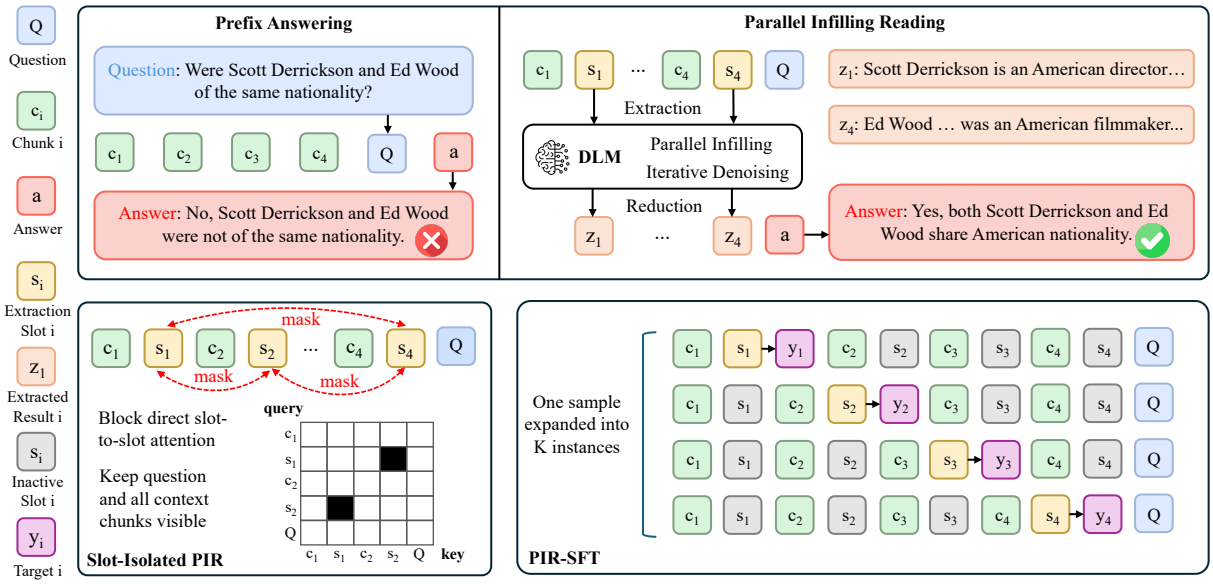

Figure 3: Overall framework of our proposed Parallel Infilling Reading approach.

evidence supervision. Specifically, given a question Q, an answer a, and a candidate document set D, we use a stronger teacher model to extract original evidence sentences from D that support the answer. We then apply deduplication and filtering to obtain a high-confidence evidence set:

$$E(Q) = \{e_1, e_2, \ldots, e_n\}. \tag{10}$$

Next, we map the sentence-level evidence back to the chunk representation used by PIR. For each chunk c_i, its local evidence set is defined as:

$$E_i = \{e \in E(Q) \mid e \subset c_i\}. \tag{11}$$

If E_i is non-empty, the target output y_i of the i-th extraction slot is the concatenation of these evidence sentences; otherwise, y_i is set to "`No relevant content found`". In this way, PIR-SFT does not merely supervise the final answer, but directly supervises the evidence extraction behavior of each slot with respect to its corresponding chunk. The full annotation prompt and filtering details are provided in Appendix A.3.

Single-Active-Slot Training. To effectively leverage the fine-grained data constructed by the above procedure, we design a single-active-slot training strategy. Given a training example with K context chunks, we construct the PIR input with K extraction slots and corresponding targets $\{y_1, \ldots, y_K\}$, and expand it into K training instances. The i-th instance keeps the complete question and the full document context, but only the i-th slot is selected as the active slot. During training, the masked span of the active slot is supervised by the target output y_i of its corresponding chunk, while the other inactive slots remain as `[MASK]` tokens in the input and do not contribute to the loss. For the i-th active slot, with supervised token positions $\mathcal{A}_i$ and noised PIR input $\tilde{x}^{(t)}_{\text{PIR},i}$, the PIR-SFT objective is:

$$\mathcal{L}_{\text{PIR-SFT}} = -\mathbb{E}_{(Q,C),i,t} \sum_{j \in \mathcal{A}_i} \log p_\theta(y_{i,j} \mid \tilde{x}^{(t)}_{\text{PIR},i}). \tag{12}$$

This training strategy decomposes multi-slot PIR into a set of more stable chunk-level infilling tasks, enabling the model to explicitly learn how to extract useful evidence while reading each chunk. At inference time, we restore the full PIR template and employ the same procedure as naive PIR. By internalizing this chunk-wise extraction capability, PIR-SFT enables the model to not rely on additional attention masks, thereby further improving inference speed.

5 Experiment

5.1 Experimental Setup

Baselines and Benchmarks. We conduct experiments on LLaDA-8B-Instruct (Nie et al., 2025) and evaluate our methods on five benchmarks: SQuAD-8K (Rajpurkar et al., 2016), HotpotQA-8K (Yang et al., 2018), HotpotQA-16K, HotpotQA-32K, and 2WikiMQA-16K (Ho et al., 2020). We compare with five representative baselines: (1) *Prefix Answering*, (2) *LongLLaDA*, (3) *Chunk-MapReduce*,

Method	SQuAD-8K		HotpotQA-8K		HotpotQA-16K		HotpotQA-32K		2WikiMQA-16K	
	SubEM	F1	SubEM	F1	SubEM	F1	SubEM	F1	SubEM	F1
AR Models										
LLaMA3.1-8B-Instruct	70.5	67.12	58.5	62.51	53.5	61.02	50.0	56.00	40.0	36.53
Qwen2.5-7B-Instruct	61.0	60.61	53.5	57.18	57.5	56.75	50.0	62.00	36.0	34.29
Training-free Methods										
Prefix Answering	30.0	15.66	53.0	24.08	31.5	16.06	18.0	9.11	23.0	11.21
LongLLaDA	68.5	25.90	69.5	30.67	64.0	27.57	32.0	11.68	41.0	17.46
Chunk-MapReduce	63.0	28.07	55.5	26.70	52.0	25.95	40.0	15.22	35.0	15.75
Retrieve-then-Reason	73.5	50.84	69.0	33.05	57.5	32.53	44.0	24.58	46.0	**33.91**
PIR w/o NTK-RoPE	77.5	46.09	56.0	29.96	36.0	19.83	22.0	7.92	34.0	18.10
PIR w/ NTK-RoPE	76.5	48.13	60.0	31.91	53.0	30.21	46.0	24.02	48.0	31.79
Slot-Isolated PIR	**80.5**	**51.74**	**70.0**	**55.16**	**64.5**	**52.36**	**54.0**	**44.92**	**53.0**	32.41
SFT Methods										
Prefix SFT	71.5	25.41	67.0	29.54	62.5	28.57	54.0	25.97	44.0	26.61
PIR-SFT	**83.5**	**63.80**	**68.5**	**58.46**	**66.0**	**54.25**	**58.0**	**47.34**	**54.0**	**33.42**

Table 1: Main results on long-context question answering benchmarks, covering both single-hop and multi-hop QA tasks. SubEM and F1 are reported for each benchmark. Bold numbers denote the best results within each DLM-based method group.

(4) *Retrieve-then-Reason*, and (5) *Prefix SFT*. We report Substring Exact Match (SubEM) and token-level F1. More details of the benchmarks and baselines are provided in Appendix B.

Implementation Details. We follow LongLLaDA (Liu et al., 2026) and set the RoPE scaling factors to 4, 14, and 31 for 8k, 16k, and 32k context lengths, respectively. For inference of PIR, we split each context into 4 chunks and set the generation length of each extraction slot to 64 tokens during the parallel evidence extraction stage, and set the maximum generation length to 16 tokens during the answer reduction stage. For training, we use the dLLM library (Zhou et al., 2026) to fine-tune LLaDA-8B-Instruct with 10k training samples. We train the model for 3 epochs with a global batch size of 32, a learning rate of 1×10^{-5}, a warmup ratio of 0.1, AdamW optimization, and a cosine learning rate schedule.

5.2 Main Results

Table 1 presents the main results on long-context QA benchmarks. First, our two PIR variants consistently outperform other baselines. Compared with other training-free methods, Slot-Isolated PIR achieves the best SubEM scores across all tasks. For example, on the most challenging HotpotQA-32K task, Slot-Isolated PIR improves SubEM by 22.0 and 8.0 points over LongLLaDA and Retrieve-then-Reason, respectively. Furthermore, introducing PIR-SFT further enhances the model's ability to solve long-context comprehension tasks, achieving the best performance on almost all tasks. These results demonstrate the effectiveness of the two improvements introduced in PIR, enabling the model to effectively extract key information from long contexts for answering questions.

Second, our two methods show greater improvements under longer contexts. On the HotpotQA dataset, compared with Retrieve-then-Reasoning, Slot-Isolated PIR improves EM scores by 1.0, 7.0, and 10.0 points at context lengths of 8K, 16K, and 32K, respectively. After applying PIR-SFT, the improvements further increase from -0.5 points at 8K to 8.5 and 14.0 points at 16K and 32K, respectively. These results strongly demonstrate the robustness of our method under long-context settings, where it consistently preserves accurate extraction of critical information from extended inputs.

Finally, the improvements in F1 are more pronounced than those in SubEM with our methods. Since DLMs tend to fill all provided mask tokens, conventional methods often generate additional content beyond the target answer, resulting in lower F1 scores. In contrast, our method first extracts key evidence and then reasons based on the extracted evidence. As a result, the model typically produces more concise and accurate answers, leading to out-

puts with better readability.

5.3 Further Analysis

Slot-Level Extraction Quality. We analyze whether our methods improve slot-level extraction using two metrics: (1) *Evidence Repetition Rate*: measuring whether different extraction slots within the same sample generate repeated evidence; (2) *Evidence Token F1*: measuring the token-level overlap between each slot output and the gold evidence sentences assigned to its corresponding chunk. As shown in Table 2, naive PIR suffers from severe slot repetition and a relatively low Evidence Token F1. After applying Slot-Isolated PIR, this phenomenon is effectively mitigated. Furthermore, employing PIR-SFT significantly reduces the evidence repetition rate while improving evidence token F1. These results indicate that our method substantially enhances the evidence extraction ability, thereby facilitating more effective extraction of relevant information from the local context.

Method	Metric	SQuAD 8K	Hotpot 8K	Hotpot 16K	Hotpot 32K	2Wiki 16K
Naive PIR	Rep.↓	83.67	71.94	66.23	42.45	43.03
	F1↑	78.11	63.81	48.44	31.26	34.27
PIR w/ NTK	Rep.↓	88.83	91.92	90.95	79.79	85.28
	F1↑	78.27	60.80	59.65	45.03	39.06
Slot-Isolated PIR	Rep.↓	59.50	52.50	47.83	10.54	20.00
	F1↑	86.73	75.47	75.23	56.66	47.40
PIR-SFT	Rep.↓	**6.22**	**23.36**	**16.98**	**8.11**	**11.72**
	F1↑	**89.85**	**79.71**	**76.25**	**74.33**	**53.58**

Table 2: Evidence quality and diversity across extraction slots. Rep. denotes the evidence repetition rate (%, ↓), and F1 denotes Evidence Token F1 (%, ↑).

Experiments on Dream-7B. To examine whether PIR generalizes beyond LLaDA, we further evaluate it on Dream-v0-Instruct-7B (Ye et al., 2025). We conduct experiments on SQuAD-8K and HotpotQA-16K, covering both single-hop and multi-hop long-context QA settings. As shown in Table 3, both Slot-Isolated PIR and PIR-SFT consistently improve over prefix-based Dream inference and other baselines. In particular, PIR-SFT substantially outperforms all other baselines. These results demonstrate the broad generalizability of our method across different DLMs.

Inference Efficiency. To investigate the inference efficiency of our method, we compare end-to-end throughput with the Retrieve-then-Reason method under different input lengths and output budgets. Specifically, we evaluate 8K, 16K, and 32K input lengths, with output budgets of 128 and 256 tokens. All the experiments are conducted on a single A800 80GB GPU. As shown in Table 4, PIR-SFT consistently achieves higher throughput than Retrieve-then-Reason owing to the parallelism of the decoding process. The advantage becomes more pronounced under longer contexts, demonstrating the advantages of our method in both effectiveness and efficiency. More detailed efficiency results are provided in Appendix D.

Method	SQuAD-8K	HotpotQA-16K
Prefix Answering	40.5	28.0
LongLLaDA	51.0	41.5
Retrieve-then-Reason	57.5	38.5
Slot-Isolated PIR	70.5	55.5
PIR-SFT	**82.5**	**59.5**

Table 3: Results on Dream-v0-Instruct-7B. We report SubEM scores on SQuAD-8K and HotpotQA-16K.

Method	Input-8K		Input-16K		Input-32K	
	128	256	128	256	128	256
RTR	1.06	0.90	0.38	0.43	0.21	0.21
PIR-SFT	**2.45**	**1.22**	**1.13**	**0.86**	**0.81**	**0.78**
Speedup	2.3×	1.4×	3.0×	2.0×	3.9×	3.7×

Table 4: Inference throughput (tokens/s) under different input lengths and output budgets. 128 and 256 denote output budgets. RTR denotes Retrieve-then-Reason.

Impact of Hyperparameters. We further study how the number and length of extraction slots affect performance. We conduct controlled experiments on SQuAD-8K and HotpotQA-16K, evaluating Slot-Isolated PIR and PIR-SFT with slot numbers $\{2, 4, 8\}$ and slot lengths $\{32, 64, 96\}$. As shown in Figure 4, too few slots generally lead to worse performance, suggesting that coarse partitioning limits evidence extraction from different context regions. Increasing the slot number improves SQuAD-8K, while HotpotQA-16K benefits more from a moderate setting. For slot length, PIR-SFT remains relatively stable across different budgets, whereas Slot-Isolated PIR is more sensitive, especially when the slot becomes too long. Overall, PIR-SFT shows stronger robustness to hyperparameter choices.

Effect of Training Strategy. We further compare single-active-slot training with parallel-slot training under the same training budget. Both methods use the same fine-grained evidence supervision, but parallel-slot training supervises all extraction

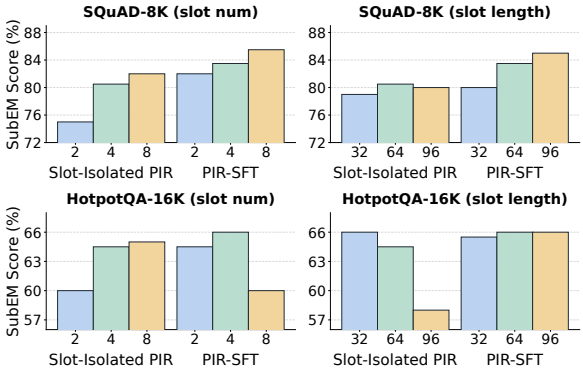

Figure 4: Impact of slot number and slot length.

Method	HotpotQA-8K	HotpotQA-16K
Parallel-Slot	63.5	54.0
Single-Active-Slot	**70.0**	**64.5**

Table 5: Effects of different training strategies.

slots within one PIR input simultaneously, while single-active-slot training activates only one slot in each training instance. As shown in Table 5, single-active-slot training consistently outperforms parallel-slot training on both HotpotQA-8K and HotpotQA-16K, indicating that focusing supervision on one chunk-slot target at a time leads to more stable extraction learning.

6 Related Work

Long-context Diffusion Language Models. Recent DLMs have demonstrated strong non-autoregressive generation abilities, but most existing studies evaluate them on relatively short contexts. To explore whether such models can handle longer inputs, LongLLaDA (Liu et al., 2026) provides a systematic analysis of long-context behavior in DLMs and shows that NTK-based RoPE extrapolation can extend their context window in a training-free manner. UltraLLaDA (He et al., 2025) further scales LLaDA to 128K contexts with diffusion-aware positional extension and masking strategies. Different from these methods that simply extend the context window, our method further enhances the model's ability to extract information from long contexts via parallel infilling reading.

Decomposed Long-context Processing. A common strategy for long-context processing is to decompose long inputs into smaller units and aggregate intermediate results. Existing methods can be divided into two categories. The first category performs explicit reasoning or extraction over split text units (Zhao et al., 2024; Zhou et al., 2025; Tang et al., 2026). The second category compresses each unit into dense or hybrid representations, and subsequent reasoning is performed over these representations rather than the full content (Jiang et al., 2024; Mu et al., 2023). Although these methods can process an entire long text under limited context windows, their independent treatment of different chunks can impair the model's ability when multi-hop reasoning or global information is required. In contrast, while PIR extracts information from each chunk individually, it retains access to the full context, ensuring a global perspective throughout the reasoning process.

Text Infilling and Diffusion-Native Prompting. Text infilling is a generation task where a model fills missing spans conditioned on surrounding contexts. It has been adopted as a pre-training objective in masked language models (Devlin et al., 2019; Lewis et al., 2020; Raffel et al., 2020), autoregressive language models (Bavarian et al., 2022; Fried et al., 2023), and the recent DLMs (Ye et al., 2025; Nie et al., 2025). Recent work has explored this paradigm on downstream tasks, such as code (Wu et al., 2026) and math (Lee et al., 2025). Unlike these span- or response-centric infilling paradigms, PIR uses infilling as a context-side reading interface by inserting masked evidence slots inside long contexts to extract and organize chunk-grounded clues before final answer generation.

7 Conclusion

In this paper, we investigate whether DLMs can better utilize their infilling ability for long-context reading. We first introduce Parallel Infilling Reading (PIR), a training-free inference method that inserts extraction slots into long contexts and enables DLMs to generate intermediate reading states through parallel infilling. Our empirical study shows that naive PIR improves over prefix-based DLM inference, but suffers from slot repetition due to cross-slot attention. To address this issue, we propose Slot-Isolated PIR to block direct slot-to-slot attention, and PIR-SFT to teach DLMs effective evidence extraction with fine-grained supervision. Experiments on long-context benchmarks demonstrate that our methods improve answer accuracy, evidence extraction, and inference throughput. These results validate infilling as an effective interface for long-context reading with DLMs, highlighting a promising direction for DLMs beyond prefix-based answering.

Limitations

Our experiments mainly focus on long-context question answering with instruction-tuned DLMs such as LLaDA and Dream. Although these benchmarks cover different context lengths and both single-hop and multi-hop settings, we have not yet evaluated PIR on broader long-context tasks such as summarization. PIR also relies on manually designed extraction and reduction templates, whose design may affect the quality of intermediate reading states. Future work can explore more general infilling templates, broader DLM backbones, and more diverse long-context scenarios.

References

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. 2024. Longbench: A bilingual, multitask benchmark for long context understanding. In *ACL (1)*, pages 3119–3137. Association for Computational Linguistics.

Mohammad Bavarian, Heewoo Jun, Nikolas Tezak, John Schulman, Christine McLeavey, Jerry Tworek, and Mark Chen. 2022. Efficient training of language models to fill in the middle. *CoRR*, abs/2207.14255.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending context window of large language models via positional interpolation. *CoRR*, abs/2306.15595.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics.

Zican Dong, Junyi Li, Jinhao Jiang, Mingyu Xu, Xin Zhao, Bingning Wang, and Weipeng Chen. 2025. Longred: Mitigating short-text degradation of long-context large language models via restoration distillation. In *ACL (1)*, pages 10687–10707. Association for Computational Linguistics.

Daniel Fried, Armen Aghajanyan, Jessy Lin, Sida Wang, Eric Wallace, Freda Shi, Ruiqi Zhong, Scott Yih, Luke Zettlemoyer, and Mike Lewis. 2023. Incoder: A generative model for code infilling and synthesis. In *ICLR*. OpenReview.net.

Guangxin He, Shen Nie, Fengqi Zhu, Yuankang Zhao, Tianyi Bai, Ran Yan, Jie Fu, Chongxuan Li, and Binhang Yuan. 2025. Ultrallada: Scaling the context length to 128k for diffusion large language models. *CoRR*, abs/2510.10481.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing A multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *COLING*, pages 6609–6625. International Committee on Computational Linguistics.

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? *CoRR*, abs/2404.06654.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. In *ACL (1)*, pages 1658–1677. Association for Computational Linguistics.

Junhoo Lee, Seungyeon Kim, and Nojun Kwak. 2025. Unlocking the potential of diffusion language models through template infilling. *CoRR*, abs/2510.13870.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, pages 7871–7880. Association for Computational Linguistics.

Huayang Li, Pat Verga, Priyanka Sen, Bowen Yang, Vijay Viswanathan, Patrick Lewis, Taro Watanabe, and Yixuan Su. 2024. Alr^2: A retrieve-then-reason framework for long-context question answering. *CoRR*, abs/2410.03227.

Tianyi Li, Mingda Chen, Bowei Guo, and Zhiqiang Shen. 2025. A survey on diffusion language models. *CoRR*, abs/2508.10875.

Xiaoran Liu, Yuerong Song, Zhigeng Liu, Zengfeng Huang, Qipeng Guo, Ziwei He, and Xipeng Qiu. 2026. Longllada: Unlocking long context capabilities in diffusion llms. In *AAAI*, pages 32186–32194. AAAI Press.

Xiaoran Liu, Hang Yan, Chenxin An, Xipeng Qiu, and Dahua Lin. 2024. Scaling laws of rope-based extrapolation. In *ICLR*. OpenReview.net.

Jesse Mu, Xiang Li, and Noah D. Goodman. 2023. Learning to compress prompts with gist tokens. In *NeurIPS*.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *CoRR*, abs/2502.09992.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2024. Yarn: Efficient context window extension of large language models. In *ICLR*. OpenReview.net.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits

of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*, pages 2383–2392. The Association for Computational Linguistics.

Jiwei Tang, Shilei Liu, Zhicheng Zhang, Qingsong Lv, Runsong Zhao, Tingwei Lu, Langming Liu, Haibin Chen, Yujin Yuan, Hai-Tao Zheng, Wenbo Su, and Bo Zheng. 2026. Read as human: Compressing context via parallelizable close reading and skimming. *CoRR*, abs/2602.01840.

Zirui Wu, Lin Zheng, Zhihui Xie, Jiacheng Ye, Jiahui Gao, Shansan Gong, Yansong Feng, Zhenguo Li, Wei Bi, Guorui Zhou, and Lingpeng Kong. 2026. Dreamon: Diffusion language models for code infilling beyond fixed-size canvas. *CoRR*, abs/2602.01326.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, Madian Khabsa, Han Fang, Yashar Mehdad, Sharan Narang, Kshitiz Malik, Angela Fan, Shruti Bhosale, Sergey Edunov, Mike Lewis, and 2 others. 2024. Effective long-context scaling of foundation models. In *NAACL-HLT*, pages 4643–4663. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, pages 2369–2380. Association for Computational Linguistics.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. 2025. Dream 7b: Diffusion large language models. *CoRR*, abs/2508.15487.

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

Zhanhui Zhou, Lingjie Chen, Hanghang Tong, and Dawn Song. 2026. dllm: Simple diffusion language modeling. *Preprint*, arXiv:2602.22661.

Zihan Zhou, Chong Li, Xinyi Chen, Shuo Wang, Yu Chao, Zhili Li, Haoyu Wang, Qi Shi, Zhixing Tan, Xu Han, Xiaodong Shi, Zhiyuan Liu, and Maosong Sun. 2025. Llm×mapreduce: Simplified long-sequence processing using large language models. In *ACL (1)*, pages 27664–27678. Association for Computational Linguistics.

A Prompt Template

A.1 PIR Prompt Template

We use the following prompt template for PIR. The long context is split into K chunks. After each chunk, we insert a local extraction prompt followed by m consecutive [MASK] tokens as the infilling slot. For readability, we omit chat-template control tokens.

PIR Prompt Template

```
– Chunk 1 –
{c_1}
Based on Chunk 1, extract the sentences that
are most relevant to the question.
Use direct quotes from the original text.
If no relevant content is found, output “No
relevant content found”.
Question: {Q}
Extracted Sentences:
[MASK] [MASK] ... [MASK]   (m masks)
– Chunk 2 –
{c_2}
Based on Chunk 2, extract the sentences that
are most relevant to the question.
Use direct quotes from the original text.
If no relevant content is found, output “No
relevant content found”.
Question: {Q}
Extracted Sentences:
[MASK] [MASK] ... [MASK]   (m masks)
...
– Chunk K –
{c_K}
Based on Chunk K, extract the sentences that
are most relevant to the question.
Use direct quotes from the original text.
If no relevant content is found, output “No
relevant content found”.
Question: {Q}
Extracted Sentences:
[MASK] [MASK] ... [MASK]   (m masks)
```

A.2 Reduce Prompt Template

We use the following template for the reduce step. The generated content from each PIR slot is formatted as chunk-level evidence, where Q denotes the question and z_i denotes the content generated by the i-th extraction slot.

Reduce Prompt Template

```
Answer the question based on the extracted
evidence.
Only output the final answer.
Information:
{z_1}, {z_2} ... {z_K}.
Question: {Q}
Answer:
[MASK] [MASK] ... [MASK]
```

A.3 Fine-Grained Evidence Annotation

To construct PIR-SFT data, we build sentence-level silver evidence supervision using a stronger teacher model. For each training sample, the teacher model receives the question Q, the gold answer a, and the candidate evidence documents D. It is asked to extract document-attributed evidence sentences that jointly support the answer. In our implementation, we use `Qwen3-32B` served by a local vLLM server, with `temperature=0.0`, `top_p=1.0`, and `max_tokens=768`.

Sentence-Level Evidence Annotation Prompt

```
You are an evidence extraction assistant.
Input:
Question: {Q}
Gold Answer: {a}
Candidate Evidence Documents: {D}
Task:
Extract sentence-level evidence from every
candidate document that supports the gold
answer to the question.
Rules:
1. Use only the candidate evidence documents.
2. Return one output item for every candidate
document.
3. Extract one or more evidence sentences for
each document.
4. Keep the original sentence whenever
possible.
5. Each returned sentence must be
self-contained and unambiguous.
6. If a sentence contains unresolved
references, minimally rewrite it by replacing
the reference with the explicit entity name.
7. Include bridge facts for multi-hop
questions.
8. Do not add unsupported facts or unnecessary
decorative details.
9. Output JSON only.
Output fields:
- document_title: ...
- document_id: ...
- evidence_sentences: [...]
- sentence: ...
- is_minimally_rewritten: true / false
```

After teacher generation, we parse the JSON output and remove duplicate evidence sentences. We validate that each item contains `document_title`, `document_id`, and a non-empty `evidence_sentences` list. We also require every returned item to match one candidate document by both title and document id, and discard samples whose parsed output is invalid or does not cover all candidate documents. The validated sentence-level evidence is then merged back into the chunked training data by sample index and document identity. For each PIR chunk, the target of its extraction slot is the concatenation of all matched evidence sentences in that chunk. If a chunk contains no matched evidence, its target is set to `No relevant content found`.

B Benchmarks and Baselines

Benchmarks. We evaluate on five benchmarks: SQuAD-8K, HotpotQA-8K, HotpotQA-16K, HotpotQA-32K, and 2WikiMQA-16K. SQuAD and HotpotQA are constructed using the RULER protocol (Hsieh et al., 2024), where SQuAD focuses on single-hop QA and HotpotQA evaluates multi-hop reasoning under different context lengths. 2WikiMQA-16K is derived from LongBench (Bai et al., 2024). We use 200 test samples for SQuAD-8K, HotpotQA-8K, and HotpotQA-16K. Due to inference cost, HotpotQA-32K is evaluated on the first 50 samples.

Baselines. We compare our methods with the following baselines.

- *Prefix Answering.* This is the standard prefix-based DLM inference baseline. The full context is provided as a prefix, and the model generates the answer only at the end of the sequence.
- *LongLLaDA.* LongLLaDA applies NTK-RoPE scaling to prefix-based DLM inference for long-context extension (Liu et al., 2026). It represents a strong training-free baseline for extending the effective context length of DLMs.
- *Chunk-MapReduce.* Following the chunk-wise divide-and-conquer paradigm of LLM×MapReduce (Zhou et al., 2025), we implement a simplified MapReduce-style baseline for DLMs. The long context is split into multiple chunks; each chunk is independently processed to produce local evidence, and the resulting chunk-level outputs are aggregated by a lightweight reducer to generate the final answer.
- *Retrieve-then-Reason.* This baseline follows the retrieve-then-reason paradigm (Li et al., 2024). It first retrieves or extracts relevant evidence from the long context, and then performs answer generation based on the selected evidence.
- *Prefix SFT.* Prefix SFT is a response-oriented supervised fine-tuning baseline. The model is fine-tuned to generate the final answer after reading the full prefix context, without explicit supervision on context-side extraction slots.

Metrics. We report Substring Exact Match (SubEM) and token-level F1. SubEM measures whether the gold answer appears as a substring of

Method	Extracted Evidence	Observation
Naive PIR	**Question:** Which French ace pilot and adventurer fly *L'Oiseau Blanc*? **Gold:** Charles Eugène c_1**:** L'Oiseau Blanc, commonly known in the English-speaking world as The White Bird, was a French Levasseur PL.8 biplane that disappeared in 1927. The aircraft was flown by French World War I aviation heroes, Charles Nungesser and François Coli. c_2**:** Charles Nungesser and François Coli. L'Oiseau Blanc, commonly known in the English-speaking world as The White Bird, was a French Levasseur PL.8 biplane that was flown by French World War I aviation heroes, Charles Nungesser and François Coli. c_3**:** Charles Nungesser and François Coli. L'Oiseau Blanc, commonly known in the English-speaking world as The White Bird, was a French Levasseur PL.8 biplane that was flown by French World War I aviation heroes, Charles Nungesser and François Coli. c_4**:** Charles Nungesser and François Coli. L'Oiseau Blanc, commonly known in the English-speaking world as The White Bird, was a French Levasseur PL.8 biplane, flown by French World War I aviation heroes, Charles Nungesser and François Coli.	The slots repeatedly extract the same surface fact about *L'Oiseau Blanc*, while missing the second-hop evidence that links Charles Nungesser to the full name Charles Eugène Jules Marie Nungesser.
Slot-Isolated PIR	c_1**:** L'Oiseau Blanc, commonly known in the English-speaking world as The White Bird, was a French Levasseur PL.8 biplane that disappeared in 1927. The aircraft was flown by French World War I aviation heroes, Charles Nungesser and François Coli. c_2**:** Charles Eugène Jules Marie Nungesser, MC, was a French ace pilot and adventurer, best remembered as a rival of Charles Lindbergh. Nungesser was a renowned ace in France. c_3**:** Charles Nungesser and François Coli flew L'Oiseau Blanc. François Coli was a French pilot and navigator best known as the one-eyed flying partner of Charles Nungesser in their doomed attempt to fly the Atlantic Ocean on the aircraft known as "L'Oiseau Blanc". c_4**:** No relevant content found.	Slot isolation reduces cross-slot repetition and improves evidence coverage. However, the extracted evidence is still verbose and contains partially redundant background information.
PIR-SFT	c_1**:** L'Oiseau Blanc was flown by French World War I aviation heroes, Charles Nungesser and François Coli. c_2**:** Charles Eugène Jules Marie Nungesser, MC, was a French ace pilot and adventurer, best remembered as a rival of Charles Lindbergh. c_3**:** No relevant content found. c_4**:** No relevant content found.	PIR-SFT extracts compact sentence-level evidence aligned with the required multi-hop reasoning chain, while avoiding both slot-level repetition and unnecessary evidence filling.

Table 6: Case study comparing the evidence extracted by naive PIR, Slot-Isolated PIR, and PIR-SFT. The extracted evidence is shown chunk by chunk, where c_i denotes the i-th context chunk. Naive PIR suffers from severe slot-level repetition, Slot-Isolated PIR improves evidence diversity but remains verbose, and PIR-SFT extracts compact sentence-level evidence that directly supports the final answer.

Method	Input-8K		Input-16K		Input-32K	
	128	256	128	256	128	256
RTR	1.06	0.90	0.38	0.43	0.21	0.21
Slot-Isolated PIR	1.16	1.08	0.44	0.43	0.22	0.22
PIR-SFT	**2.45**	**1.22**	**1.13**	**0.86**	**0.81**	**0.78**

Table 7: Detailed inference throughput (tokens/s) under different input lengths and output budgets. The columns 128 and 256 denote output budgets. RTR denotes Retrieve-then-Reason.

the model prediction, while F1 measures token-level overlap between the prediction and the reference answer.

C Case Study

We provide a case study in Table 6 to compare the evidence extracted by different PIR variants. The results show that PIR-SFT produces more compact and less redundant sentence-level evidence than naive PIR and Slot-Isolated PIR.

D Inference Efficiency Analysis

We provide detailed throughput results in Table 7. All experiments are conducted on a single A800 80GB GPU, and throughput is measured in tokens/s under different input lengths and output budgets.