Attention Entropy is a Key Factor: An Analysis of Parallel Context Encoding with Full-attention-based Pre-trained Language Models

Anonymous ACL submission

Abstract

Large language models have shown remarkable performance across a wide range of language tasks, owing to their exceptional capabilities in context modeling. The most commonly used method of context modeling is full self-attention, as seen in standard decoder-only Transformers. Although powerful, this method can be inefficient for long sequences and may overlook inherent input structures. To address these problems, an alternative approach is parallel context encoding, which splits the context into sub-pieces and encodes them parallelly. Because parallel patterns are not encountered during training, naively applying parallel encoding leads to performance degradation. However, the underlying reasons and potential mitigations are unclear. In this work, we provide a detailed analysis of this issue and identify that unusually high attention entropy can be a key factor. Furthermore, we adopt two straightforward methods to reduce attention entropy by incorporating attention sinks and selective mechanisms. Experiments on various tasks reveal that these methods effectively lower irregular attention entropy and narrow performance gaps. We hope this study can illuminate ways to enhance context modeling mechanisms.

1 Introduction

Large language models (LLMs) have demonstrated exceptional capabilities across various language tasks (Achiam et al., 2023; Dubey et al., 2024). A key factor contributing to this success is their remarkable ability of context modeling. This capability forms the basics of instruction following (Ouyang et al., 2022; Bai et al., 2022) and in-context learning (ICL; Brown et al., 2020; Dong et al., 2024), enabling LLMs to comprehend contexts effectively. Consequently, LLMs can solve tasks directly when provided with appropriate prompts (Liu et al., 2023).

To model contexts, most LLMs adopt a similar architectural design: an auto-regressive decoder-only Transformer with full self-attention (Vaswani et al., 2017; Radford et al., 2019). This architecture does not assume contextual independence, allowing each token to attend to all previous tokens. While powerful and flexible, this design is not without concerns. First, full attention requires computational complexity that scales quadratically with the input sequence length. This poses challenges for long sequence processing and necessitates more efficient alternatives (Tay et al., 2022). Additionally, in many applications, contexts or prompts exhibit *natural parallel structures*, consisting of independent sub-pieces, such as documents in retrieval-augmented generation (RAG; Lewis et al., 2020) and demonstrations in ICL. It is intuitive to leverage these structures more effectively.

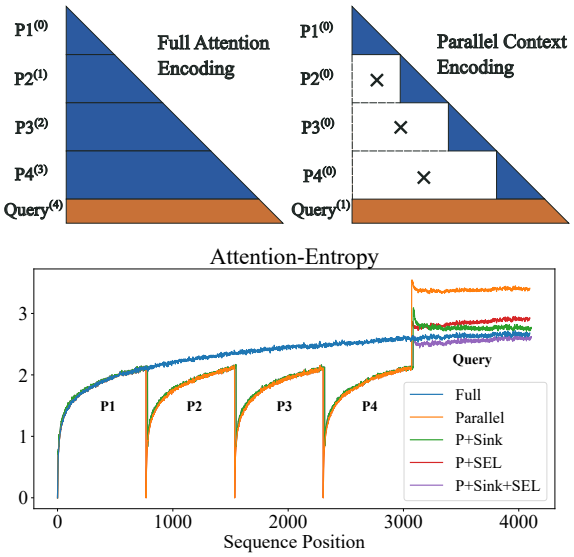

Figure 1: An overview. The upper part illustrates the full attention and parallel context encoding schemes (superscripts denote position encoding), while the lower part shows parallel encoding leads to irregularly high attention entropy for the query tokens. We explore two methods to reduce entropy: "Sink" means adding shared attention sinks, and "SEL" means selective attention.

To enhance the efficiency of context encoding and leverage the input structures, a natural strategy is to divide the context into sub-pieces, encode each one in parallel, and then concatenate them for final use. Figure 1 illustrates the differences between full attention encoding and parallel context encoding. Compared to full encoding, the parallel approach can reduce the computational complexity since each sub-piece does not interact with others during the context encoding phase, and the parallel input structures are directly utilized for context splitting. However, mainstream LMs typically rely on full attention and haven't been trained with parallel contexts, posing the question of *whether parallel context encoding is compatible with full-attention-based pre-trained LMs*. While specialized fine-tuning can ensure compatibility, it can also be computationally costly (Yen et al., 2024a; Sun et al., 2024; Lu et al., 2024). Recent studies have explored settings that do not require additional fine-tuning but these are limited to specific scenarios, such as restricted numbers of context windows (Ratner et al., 2023) or specific tasks like ICL (Hao et al., 2022) or RAG (Merth et al., 2024). In contrast, through detailed evaluations over various tasks, we provide more comprehensive analyses of this question, showing the connections between irregular attention entropy and the final performance.

We conduct experiments on a variety of language tasks, including language modeling (LM), ICL, RAG and synthetic tasks. Through fair and direct comparisons between full-attention and parallel encoding schemes, we demonstrate that naively applying parallel encoding results in significant performance declines. By analyzing the attention patterns of both schemes, we find that parallel encoding leads to higher attention entropy on the final query tokens (Figure 1 shows a typical example). Furthermore, we discover strong correlations between attention entropy and model performance, suggesting that attention entropy can be an indicator of irregular performance. To address this, we adopt two straightforward methods to reduce attention entropy: *attention sinks* (Xiao et al., 2024), which adds a shared prefix to the context that each sub-piece can attend to, and *selective attention*, which incorporates a hard selection mechanism into the attention operation. Experimental results show that both methods can reduce the irregular attention entropy and mitigate the performance gaps, verifying our hypothesis. Additionally, we provide a detailed analysis of how different selective attention choices affect performance across various tasks. We hope that our analysis could offer insights into exploring alternative context-modeling mechanisms beyond the full attention scheme.

2 Experimental Settings

2.1 Parallel Context Encoding

In a vanilla Transformer-based LM, to encode a context sequence of N tokens (assuming the use of a decoder-only model with causal masks), each token needs to attend to all preceding tokens in the context. Consequently, we need to calculate the attention scores for $\frac{1}{2} \cdot N(N+1)$ token pairs. With parallel context encoding, we split the context into P sub-pieces.[1] In this scheme, each piece is encoded separately, and tokens within one piece do not attend to tokens in other pieces. Assuming that we evenly split the context for simplicity, the token-pair calculation requirement is $P \cdot \frac{1}{2} \cdot \frac{N}{P}(\frac{N}{P}+1) = \frac{1}{2P} \cdot N(N+P)$, which is approximately $\frac{1}{P}$ of the computations needed in full attention. Therefore, the more pieces the context is split into, the more computational savings can be achieved.

The parallel scheme is intuitive in many applications, such as RAG and ICL. This is because each piece, such as a document in RAG or a demonstration in ICL, is self-contained and does not require additional information in its encoding phase. The main phase where we need to check full contexts and aggregate information across pieces is the query-encoding phase. At this stage, we can let the querying tokens attend to the all preceding tokens to gather information.

While there can be minor variations, the basic methodology for parallel context encoding remains largely the same in previous research. Following Ratner et al. (2023), we provide a brief introduction of the two main modifications to full attention: *position encoding* and *attention masking*.

For *position encoding*, each piece is parallel to each other and uses its own position counting mechanism. If the pieces have different lengths, we take the maximum length as the target context length and evenly distribute the position encoding of the tokens within each piece accordingly.[2] For ex-

[1] We refer to the number of sub-pieces as *parallel degree*, which is one of the main variables examined in this study.

[2] We explore models that utilize RoPE (Su et al., 2024), which allows for the assignment of real-valued position IDs. There can be other options for position encoding, including using the harmonic mean as the target length (Merth et al., 2024)

	LM (PPL↓)			ICL (Acc↑)			RAG (SubEM↑)			Synthetic (SubEM↑)		
	4K	8K	16K	4K	8K	16K	4K	8K	16K	4K	8K	16K
Full	5.54	5.35	5.19	55.20	66.00	72.20	61.25	60.25	60.25	99.88	99.50	97.25
P=2	5.83	5.66	5.47	50.80	63.60	70.20	61.50	59.50	57.50	93.81	94.81	95.25
P=4	6.29	6.16	6.04	36.40	57.20	67.80	59.25	50.75	52.50	79.19	81.56	82.44
P=8	6.91	6.92	6.96	29.20	44.40	60.20	53.50	48.75	44.50	25.94	41.00	41.44
P=16	7.69	7.97	8.54	21.00	34.00	46.40	49.00	41.75	40.00	3.38	2.19	2.00
P=32	8.54	9.24	10.87	10.80	17.40	33.60	45.00	39.25	35.25	0.31	0.00	0.00
P=64	9.35	10.46	13.18	5.00	10.80	19.80	45.00	33.00	26.75	0.00	0.00	0.00

Table 1: Comparisons between full-attention and naive parallel encoding with LLAMA-3.1-8B (results are macro-averaged over all sub-tasks). Here, "P" indicates the parallel degree (the number of context sub-pieces). For each task, we also vary the total sequence lengths (considering 4K, 8K and 16K).

ample, assume we have three context pieces with lengths of $L1$, $L2$, and $L3$. With full attention, we need to arrange them sequentially and assign positions ranging from 0 to $L1 + L2 + L3$ to all the tokens. With parallel encoding, we no longer need a specific order among different pieces; each piece independently counts its own tokens' positions starting from 0 to the target length.

For *attention masking*, we design special attention masks in accordance with the parallel encoding scheme. Each token within a context piece is restricted to attend only to the preceding tokens within this piece but not to other pieces. However, the final query tokens can attend to all preceding tokens across all context pieces to aggregate information. This approach results in inherently sparse attention calculations, for which sparse attention tools, such as FlexAttention[3] and block-sparse attention in FlashAttention (Dao et al., 2022), can be used to enhance efficiency.

2.2 Setups

Task. We experiment with a variety of language tasks to evaluate the influence of parallel context encoding, including LM, ICL, RAG and synthetic recall tasks. For LM, we use the PG19 (Rae et al., 2020) and Proof-Pile (Azerbayev et al., 2023) datasets for evaluation. For the remaining tasks, we take the corresponding datasets from the HELMET benchmark (Yen et al., 2024b) and follow its processing protocols. Specifically, these include TREC-coarse and TREC-fine (Li and Roth, 2002), BANKING77 (Casanueva et al., 2020), CLINC150 (Larson et al., 2019) and NLU (Liu et al., 2019) for ICL; Natural Question (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), HotpotQA (Yang et al., 2018) and PopQA (Mallen et al., 2023) for RAG; and three typical needle-in-a-haystack tasks (Kamradt, 2023) from RULER (Hsieh et al., 2024) as well as a JSON retrieval task (Liu et al., 2024a) for synthetic recall.

Evaluation. For all tasks, we assume that an input instance consists of a context and a query. The context can be further split into sub-pieces, for which we can apply parallel encoding, and the query can always attend to all previous contexts. For non-LM tasks, this scheme is natural: each instance already contains a query and a context consisting of a collection of items (documents in RAG, demonstrations in ICL, and haystack items in synthetic recall). Note that for parallel encoding, we can group multiple items into one sub-piece when we want a parallel degree (i.e., the number of parallel sub-pieces) that is smaller than the number of available items. For LM, we simulate this scheme by designating the final 1K tokens in a text segment as the query. The preceding tokens are considered as the context and are evenly divided into sub-pieces for parallel encoding. To evaluate the performance, we measure the perplexity (PPL) of the query tokens for LM. For other tasks, we follow HELMET and measure substring exact match for RAG and synthetic recall, and accuracy for ICL by comparing the model's generated output (with greedy decoding) to the gold answers.

Model. We use LLAMA-3.1-8B as the primary model in our main experiments. Results from other models, including the INSTRUCT version, MISTRAL-7B-V0.3 and QWEN2-7B, exhibit similar overall trends and are detailed in Appendix A. These models share a similar architecture design with RoPE-based positional encoding, which is

or retaining natural integer counting (Ratner et al., 2023). Our choice is based on its overall good performance in preliminary experiments with our settings.

[3] https://pytorch.org/blog/flexattention/

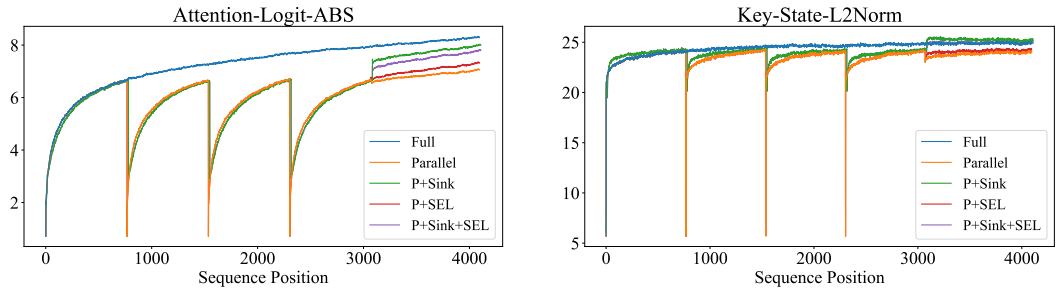

Figure 2: The scales of attention logits (averaged absolute values) and key states (L2 norm) with different methods. The irregularities of these scales may explain why attention entropy is higher with parallel context encoding.

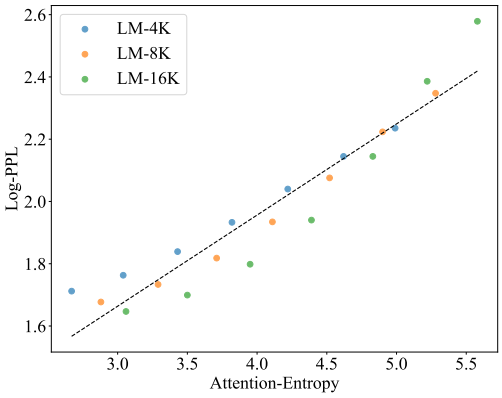

Figure 3: An illustration of the correlations between model performance and attention entropy (with the LM task and LLAMA-3.1-8B).

adaptable and facilitates modifications for parallel encoding. Our experiments utilize the pre-trained models as they are, without any fine-tuning. However, modifications to the internal attention layers are necessary, which is why we cannot evaluate closed-source LLMs.

3 Attention Entropy as an Indicator of Irregularities

Naively applying parallel encoding leads to poorer performance. Table 1 presents the main results of directly applying parallel context encoding to different tasks. Across all tasks, the direct application leads to worse results, and the performance degradation becomes more pronounced as the parallel degree increases. Notably, we can observe a dramatic decline for synthetic recall tasks: from nearly perfect accuracy with full attention to nearly complete failure when the context is split into tens of sub-pieces. This outcome is somewhat expected, as LLMs are trained with full attention and are thus unaccustomed to parallel contexts. This suggests that there could be some irregularities in the internal states of LLMs that are likely causing this failure.

Attention entropy can be an indicator of irregularities. Inspired by recent studies in LLM length extrapolation (Han et al., 2024), we examine and compare the attention values of different context encoding schemes. In length extrapolation, it is intuitive that longer sequences result in higher attention entropy values. Interestingly, we also observe irregularly high attention entropy for the query tokens when attending to parallel contexts. The lower part of Figure 1 shows a typical example: it shows the averaged attention entropy values for the PG19 LM task (4K) with LLAMA-3.1-8B[4] and a parallel degree of four. It demonstrates that when attending to parallel contexts, the attention entropy is much higher than that with vanilla full attention. Higher entropy usually denotes a higher level of uncertainty and confusion, which might explain why LLMs struggle to accurately retrieve information from the parallelly encoded contexts. Figure 3 illustrates the relationship between attention entropy and model performance, revealing strong correlations (PEARSONR$\approx$0.95) between them.

Irregularly high entropy can be attributed to irregularities in hidden state scales. We further investigate[5] the causes of irregularly high attention entropy. Firstly, we examine the scales of attention logits – the input to the attention softmax operation. As shown in the left sub-figure of Figure 2, we again find irregularities: the averaged absolute values of attention logits are smaller with parallel encoding. To examine what causes the irregu-

[4]Results are averaged over all layers and heads. Results with other models and tasks show similar patterns.

[5]For this analysis, we again average over all the layers and heads. Note that, for these scales, we observe larger variations among different heads than those in the entropy analysis. Nevertheless, we think that the averaged results can still meaningfully provide an overall explanation.

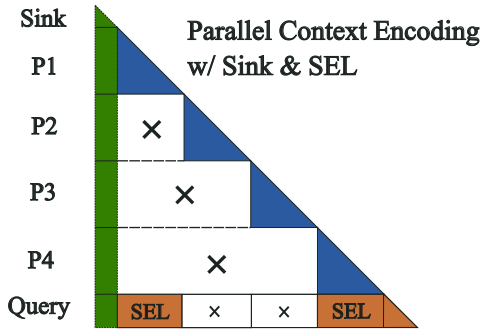

Figure 4: Illustrations of our methods to reduce attention entropy: adding shared attention sinks (Sink) and adopting selective attention (SEL).

lar logit scales, we further inspect the key states – the inputs to the MATMUL operation that produces the attention logits. As illustrated in the right sub-figure of Figure 2, the norm of the key states generally increases along the sequence dimension. With parallel context encoding, where the context pieces are encoded individually, the key states have smaller norms than those in full attention. Especially, the initial tokens in each piece, which are known as sink tokens (Xiao et al., 2024), have dramatically smaller norms (Gu et al., 2024). While it would be interesting to further investigate the cause of the irregular hidden state patterns, we find that this involve complex interactions with various Transformer layers, such as LayerNorm and MLP; a complete explanation of this phenomenon would require a deeper understanding of the underlying working mechanisms of Transformers, which we leave to future exploration.

4 Reducing Entropy with Attention Sinks and Selective Attention

4.1 Methods

Our prior analysis indicates a strong correlation between model performance and attention entropy. However, *correlation does not imply causation*. To investigate whether the irregular attention entropy is a key factor of performance degradation, we adopt two straightforward methods to adjust the attention entropy, attention sinks and selective attention, as depicted in Figure 4.

4.1.1 Attention Sinks

Recent studies on attention sinks have demonstrated that initial tokens significantly influence the internal dynamics of Transformers (Xiao et al., 2024; Han et al., 2024; Gu et al., 2024). As shown in Figure 2, we also observe that the sinking tokens exhibit abnormal hidden state scales, potentially leading to irregular attention entropy. When naively applying parallel context encoding, each sub-piece contains its own sinks, which are subsequently attended to by later query tokens. The model has never encountered such multi-sink patterns in LM training and thus produces irregular hidden states. To mitigate this problem, we prepend a shared prefix to all the context sub-pieces to eliminate attention sinks inside each sub-piece. Interestingly, preliminary experiments indicate that the specific content of the shared prefix is not crucial; even adding tokens of linebreaks can be effective, indicating that their main functionality is to absorb unneeded attention values. Without loss of generality, we manually write simple instructions[6] as the shared prefixes.

The impact of incorporating shared attention sinks can be examined by analyzing LM's internal states. As shown in Figure 2, attention sinks can avoid the extremely irregular tokens in each sub-piece (the original sink tokens) and lead to higher attention logit values, which lead to lower attention entropy as depicted in Figure 1. As discussed in the following subsection, this can indeed enhance performance, suggesting that shared attention sinks can help the model to be more familiar with the hidden state patterns of parallel context encoding.

4.1.2 Selective Attention

An alternative method to reduce attention entropy is to directly sharpen the attention distribution through hard selection. Specifically, we group the context tokens according to the splitting of the parallel sub-pieces. A sub-piece score is then calculate for each group, followed by a top-K selection process for each attention operation. For instance, in the scenario of four context pieces as depicted in Figure 4, we select the top-2 scored sub-pieces and exclude the remaining two from the attention calculation. As shown in Figure 1, this selective mechanism can directly reduce entropy.

The overall procedure is outlined in Algorithm 1. It can be easily understood by examining the shapes of the intermediate tensors.

- **Input.** The input attention probability tensor p_{in} has the shape of $[N_{layer}, N_{head}, L_{query}, L_{key}]$.
- **Grouping.** We first compute the group score

[6] For LM, we use *"Given the following partial context, predict the next sequence of words:"*; for other tasks, we use *"Given the following contexts, answer the final question accordingly:"*.

Algorithm 1 Selective Attention.

Input: Original attention probability tensor p_{in}.
Output: Modified attention probability tensor p_{out}.

$s_{group} \leftarrow$ group_key(p_{in}) ▷ Obtain grouped scores
$i_{sel} \leftarrow$ top_k(s_{group}) ▷ Group selection
$m \leftarrow$ expand_mask(i_{sel}) ▷ Expand SEL mask
$p_m \leftarrow p_{in} \cdot m$ ▷ Apply mask
$p_{out} \leftarrow p_m / p_m.sum(-1)$ ▷ Re-normalization
return p_{out}

for each sub-piece along the "Key" dimension: each group has a piece of attention probabilities, which are reduced into one group score. We use the sum of the top-5 values[7] as the reduction function, which is found to be better than using sum or average. Assuming there are P sub-pieces, the final group score s_{group} will have the shape of $[N_{layer}, N_{head}, L_{query}, P]$.

- **Selection.** The selection is performed along the group dimension, where only the top-K[8] scored groups are selected as valid. We obtain the selected group indexes i_{sel}, which has the shape of $[N_{layer}, N_{head}, L_{query}, K]$.
- **Masking.** The selected indexes are expanded to obtain the selection mask over the original tokens. Tokens within parallel contexts that do not belong to any selected groups will be masked out. This mask m has the same shape as p_{in}.
- **Output.** Finally, the mask m is applied to the input probability tensor, and the final output attention probability tensor is obtained after a final re-normalization step to ensure that each row sums up to one.

Between the grouping and selection step, an optional reduction operation can be performed to aggregate information among tokens, heads or even layers. For example, if aggregating over the query-token dimension, s_{group} is reduced from $[N_{layer}, N_{head}, L_{query}, P]$ to $[N_{layer}, N_{head}, 1, P]$. This reduction is useful in scenarios where the most relevant information comes from the same sub-piece for all tokens in the current query. If aggregating over heads, it can help to identify the most salient information-seeking head, such as the retrieval head (Wu et al., 2024). A more aggressive reduction can be performed across the first three dimensions, reducing the group score to $[1, 1, 1, P]$, which is exactly the same as a retrieval procedure. Notice that if we choose the layer dimension for aggregation, we need to forward the model twice since attention scores at later layers are not available when calculating previous layers; for other dimensions, the selective attention modification can be applied immediately after each attention score is calculated. We again use the sum of top-5 values as the reduction function to identify the most salient attention scores.

We do not apply aggregation for the LM task since it often requires diverse information from their contexts; for other tasks where there are clear queries and information sources, we reduce on the head and query dimensions by default, which is found to perform well overall. We provide further analyses on the specifications of selection attention for different tasks in §4.3.1.

[7]Preliminary experiments indicate that the results are not very sensitive to the number of top values used in this step.

[8]We choose K=2 as the default value since earlier results (see Table 1) demonstrate that using two parallel contexts does not significantly impact the outcomes.

4.2 Main Results

Figure 5 illustrates the effectiveness of the entropy reduction methods (with LLAMA-3.1-8B and 8K lengths). The overall trends are consistent across different tasks. First, both shared sink tokens and selective attention can reduce attention entropy and enhance performance compared to the naive parallel scheme, especially with higher parallel degrees. Additionally, these two methods impact attention entropy differently: with sinks, the entropy is lower than the naive scheme, but still grows larger than that of full attention (P=1); with selective attention, the entropy decreases and can sometimes become even lower than that of full attention. Lastly, the benefits of these methods vary depending on the task. Selective attention is more helpful for RAG and synthetic recall tasks, which is intuitive because of the retrieval nature of these tasks. On the other hand, attention sinks seem to be more beneficial for ICL tasks, since these tasks may need information from more demonstration examples than our default selective top-K value (K=2). Combining both techniques offers a balanced approach and yields overall effective performance.

4.3 Analyses

4.3.1 Variations on Selection Attention

We provide detailed ablation studies on various choices in the selection attention procedure. Since there are no clear query tokens in the LM task, our analysis primarily focuses on the other three tasks. We consider a typical scenario as the case study:

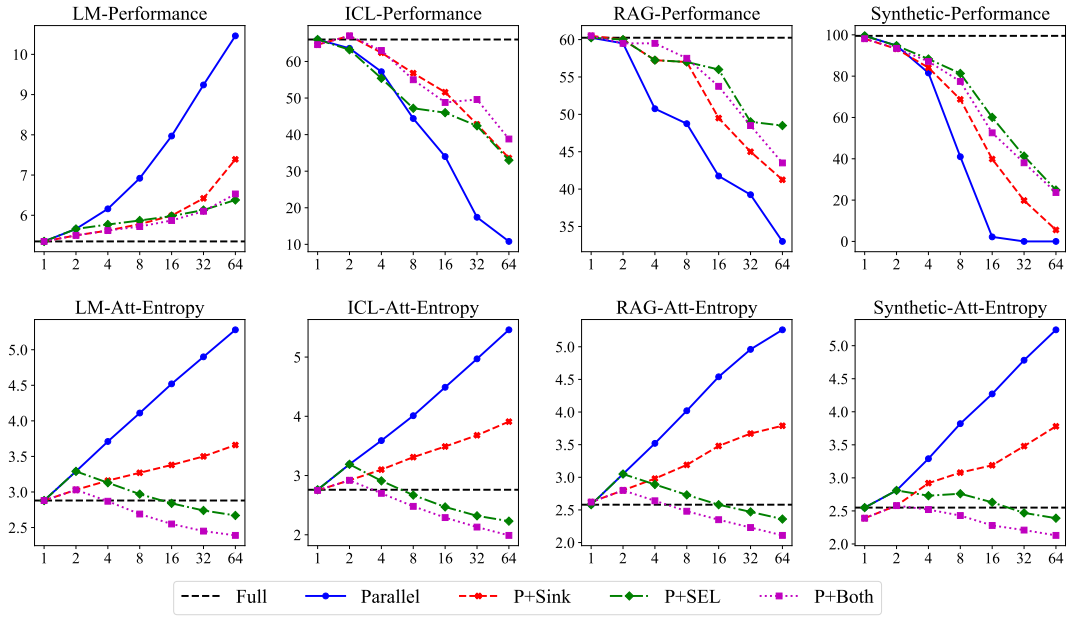

Figure 5: Results of entropy reduction methods (with LLAMA-3.1-8B and 8K lengths). The x-axis denotes the parallel degree P. The upper figures illustrate the model's performance: PPL for LM (the lower the better) and Accuracy or SubEM for other tasks. The lower figures denote the averaged attention entropy over the query tokens. Additional results for more models and settings are presented in Appendix A.

using LLAMA-3.1-8B and 8K lengths, and adopting a difficult parallel degree of P=64. Firstly, we start with our default setting of aggregating over the Head and Token dimensions (denoted as "HT"), and vary the K value in sub-piece top-K selection process. The results indicate that the optimal setting varies by task: synthetic recall tasks benefit from a small K value, since they require precise information from a few pieces, while ICL and RAG perform better with slightly larger K values, since additional context information can be helpful. Next, we examine different ways of information aggregation (with TopK=5 for ICL and RAG, and TopK=2 for synthetic tasks). Once again, different tasks exhibit distinct patterns: aggregating over all layers, as in a retrieval setting, yields the best results for RAG, layer-wise selection is more effective for synthetic tasks, and query-level selection is not crucial for ICL. While a universal and consistent method that performs well across all tasks would be ideal, achieving this may be difficult and costly. One advantage of our method is its flexibility, allowing dynamic adjustment of configurations to suit the specific nature of each task.

4.3.2 Effects of Value-only Parallel Encoding

In our experiments, we mainly examine the attention patterns and methods to reduce attention entropy. In parallel context encoding, the value states are also influenced. To investigate the impact of value states, we consider an oracle setting[9] where we replace the key states with those from full attention encoding; in this way, we have a mixed setting of value-only parallel encoding. Figure 6 illustrates the results. Except for LM, using oracle key states does not always perform better than our methods, indicating that value states also play an important role in contextual encoding.

	ICL	RAG	Synthetic
TopK=1	26.00	45.75	21.56
TopK=2	33.00	48.50	**24.88**
TopK=5	**36.00**	**48.75**	14.69
TopK=10	28.60	44.50	5.25
No Aggr.	35.40	42.75	17.62
Aggr.=T	**36.20**	45.00	21.00
Aggr.=HT	36.00	48.75	**24.88**
Aggr.=LHT	22.40	**49.50**	17.31

Table 2: Ablation studies on selection attention (with LLAMA-3.1-8B, 8K lengths and P=64). "TopK" denotes how many sub-pieces to select for each attention, "Aggr." means the dimensions on which we apply aggregation (Layer, Head or Token).

5 Related Work

Parallel Context Modeling. Recent research has explored parallel context encoding for various tasks.

[9] We've also tried only replacing value states, which leads to significantly worse and meaningless results.

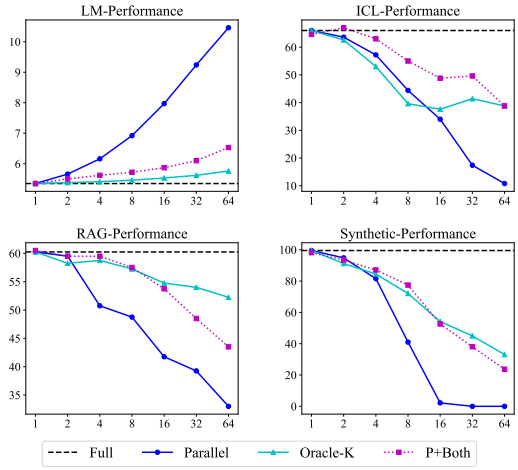

Figure 6: Performance with the oracle setting of value-only parallel encoding.

Ratner et al. (2023) present parallel context window to extend LLMs for handling longer contexts, which is beneficial for ICL and RAG tasks. Similarly, Hao et al. (2022) scale ICL to accommodate thousands of demonstrations with a similar approach. Yen et al. (2024a) train an additional context encoder and cross-attention layers to achieve enhanced context encoding, albeit at a higher computational cost. Furthermore, parallel encoding has been applied to RAG (Merth et al., 2024; Sun et al., 2024; Lu et al., 2024), where the retrieved documents are naturally parallel to each other. Beyond encoding, the decoding process can be also made parallel, as explored in non-autoregressive generation (Stern et al., 2018; Ghazvininejad et al., 2019) and more efficient LLM prompting techniques (Ning et al., 2024).

Efficient Attention. In addition to parallel context encoding, there has been considerable work on the topics of efficient attention (Tay et al., 2022). Adopting sparse attention patterns is a typical approach that selects certain tokens in the attention mechanism with either fixed (Child et al., 2019; Beltagy et al., 2020) or learned (Kitaev et al., 2020; Roy et al., 2021; Gupta et al., 2021) patterns. Parallel context encoding can be viewed as a special form of sparse attention, which enhances block sparsity. Another line of work focuses on efficient approximation of full attention using linear attention techniques (Katharopoulos et al., 2020; Choromanski et al., 2021; Peng et al., 2021). Recently, with the advent of LLMs, there has been a renewed interest in efficient attention mechanisms for Transformer models to reduce computational and memory costs. Prompt or KV-cache compression techniques have been widely investigated, and the approaches mainly include training special compressing tokens (Mu et al., 2023; Chevalier et al., 2023; Ge et al., 2024b; Qin et al., 2024; Mohtashami and Jaggi, 2024) or dynamically selecting tokens at inference time (Zhang et al., 2023; Liu et al., 2024b; Ge et al., 2024a; Li et al., 2024). Our attention selection approach shares similar spirits to the latter strategies, though we perform the selection over context blocks.

Attention Analysis. Since the introduction of self-attention in Transformers, analyzing the roles the attention mechanism plays has been a popular topic (Clark et al., 2019; Jain and Wallace, 2019; Serrano and Smith, 2019; Wiegreffe and Pinter, 2019; Bibal et al., 2022). The most relevant work to this study includes findings on attention sinks and specialized attention heads. Attention sinks refer to the initial tokens that attract most of the attention weights in many heads, and they have been utilized to extend LLMs to longer context lengths (Xiao et al., 2024; Han et al., 2024; Gu et al., 2024). These works also inspire our analyses on attention entropy and hidden state norms. Additionally, it has also been shown that there can be specialized attention heads that perform special functions, such as syntactic heads for encoding syntactical relations (Clark et al., 2019), retrieval heads for collecting information from long contexts (Wu et al., 2024), and induction heads that may constitute the mechanism for ICL (Olsson et al., 2022). Our reduction operations in selective attention are also based on the hypothesis that there is a small portion of information-seeking heads that can collect the most salient features from the contexts.

6 Conclusion

In this work, we present a detailed analysis of parallel context encoding for full-attention-based LMs without any fine-tuning. We demonstrate that naively applying parallel encoding leads to noticeably worse performance, particularly as the parallel degree increases. Through our analyses, we discover a strong correlation between irregularly high attention entropy and performance degradation. We adopt two approaches to reduce the entropy, which can help mitigate the performance gaps. We hope that our analyses and results can shed light on a deeper understanding and improvement of attention mechanisms.

Limitations

This work has several limitations. First, we primarily use the pre-trained LM as it is without applying any fine-tuning. Clearly, fine-tuning could mitigate the irregularities in parallel encoding and enhance performance. However, it will bring extra computational costs, and selecting appropriate fine-tuning datasets would require careful consideration to maintain the model's general capabilities. Second, we mainly focus on performance analyses in this work, while leaving efficient implementation and related analyses to future work, which would require kernel-level implementations to achieve speed improvements. Finally, we have not found a universal and consistent method to fully address the performance gaps between full attention and parallel context encoding schemes. Further investigation and the incorporation with lightweight fine-tuning may help to close these gaps.

Ethics Statement

This research primarily concentrates on analyses of language models. Consequently, we have not implemented any extra aggressive filtering techniques on the text data beyond the preprocessing done by the original dataset sources. We have also employed open-source language models in their existing form, without further addressing aspects such as enhancing safety and debiasing. As a result, the text data and models we used may contain issues related to offensiveness, toxicity, fairness, or bias that we have not identified, as these concerns are not the main focus of our study. Apart from these considerations, we do not foresee any additional ethical concerns or risks associated with our work.

References

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

Zhangir Azerbayev, Edward Ayers, and Bartosz Piotrowski. 2023. Proofpile: A pre-training dataset of mathematical texts.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

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

Adrien Bibal, Rémi Cardon, David Alfter, Rodrigo Wilkens, Xiaoou Wang, Thomas François, and Patrick Watrin. 2022. Is attention explanation? an introduction to the debate. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3889–3900, Dublin, Ireland. Association for Computational Linguistics.

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

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

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, Singapore. Association for Computational Linguistics.

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

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. 2021. Rethinking attention with performers. In *International Conference on Learning Representations*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu,

Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A survey on in-context learning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1107–1128, Miami, Florida, USA. Association for Computational Linguistics.

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

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. 2024a. Model tells you what to discard: Adaptive KV cache compression for LLMs. In *The Twelfth International Conference on Learning Representations*.

Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2024b. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations*.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6112–6121, Hong Kong, China. Association for Computational Linguistics.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. 2024. When attention sink emerges in language models: An empirical view. *arXiv preprint arXiv:2410.10781*.

Ankit Gupta, Guy Dar, Shaya Goodman, David Ciprut, and Jonathan Berant. 2021. Memory-efficient transformers via top-k attention. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 39–52, Virtual. Association for Computational Linguistics.

Chi Han, Qifan Wang, Hao Peng, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. 2024. LM-infinite: Zero-shot extreme length generalization for large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3991–4008, Mexico City, Mexico. Association for Computational Linguistics.

Yaru Hao, Yutao Sun, Li Dong, Zhixiong Han, Yuxian Gu, and Furu Wei. 2022. Structured prompting: Scaling in-context learning to 1,000 examples. *arXiv preprint arXiv:2212.06713*.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. 2024. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

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

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR.

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

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

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

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. 2024. SnapKV: LLM knows what you are looking for before generation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024a. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2019. Benchmarking natural language understanding services for building conversational agents. *ArXiv*, abs/1903.05566.

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. 2024b. Scissorhands: Exploiting the persistence of importance hypothesis for llm kv cache compression at test time. *Advances in Neural Information Processing Systems*, 36.

Songshuo Lu, Hua Wang, Yutian Rong, Zhi Chen, and Yaohua Tang. 2024. Turborag: Accelerating retrieval-augmented generation with precomputed kv caches for chunked text. *arXiv preprint arXiv:2410.07590*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Thomas Merth, Qichen Fu, Mohammad Rastegari, and Mahyar Najibi. 2024. Superposition prompting: Improving and accelerating retrieval-augmented generation. *arXiv preprint arXiv:2404.06910*.

Amirkeivan Mohtashami and Martin Jaggi. 2024. Random-access infinite context length for transformers. *Advances in Neural Information Processing Systems*, 36.

Jesse Mu, Xiang Lisa Li, and Noah Goodman. 2023. Learning to compress prompts with gist tokens. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Xuefei Ning, Zinan Lin, Zixuan Zhou, Zifu Wang, Huazhong Yang, and Yu Wang. 2024. Skeleton-of-thought: Prompting LLMs for efficient parallel generation. In *The Twelfth International Conference on Learning Representations*.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. 2021. Random feature attention. In *International Conference on Learning Representations*.

Guanghui Qin, Corby Rosset, Ethan Chau, Nikhil Rao, and Benjamin Van Durme. 2024. Dodo: Dynamic contextual compression for decoder-only LMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9961–9975, Bangkok, Thailand. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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

Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. Parallel context windows for large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6383–6402, Toronto, Canada. Association for Computational Linguistics.

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

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. 2018. Blockwise parallel decoding for deep autoregressive models. *Advances in Neural Information Processing Systems*, 31.

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

East Sun, Yan Wang, and Tian Lan. 2024. Block-attention for efficient rag. *arXiv preprint arXiv:2409.15355*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

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

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

Howard Yen, Tianyu Gao, and Danqi Chen. 2024a. Long-context language modeling with parallel context encoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2588–2610, Bangkok, Thailand. Association for Computational Linguistics.

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

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. 2023. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710.

A Additional Results

In the appendix, we provide several additional results:

- Table 3 and 4 show the main results of parallel context encoding using MISTRAL-7B-V0.3 and QWEN2-7B, whose patterns are similar to LLAMA-3.1-8B as shown in Table 1.
- Figure 7 presents the correlations between model performance and attention entropy for other tasks, and the patterns are similar to those in the LM task as shown in Figure 3.
- Figure 8, 9, 10 and 11 illustrate more results of entropy reduction methods in different settings. The overall trends are similar to those in Figure 5.

	LM (PPL↓)			ICL (Acc↑)			RAG (SubEM↑)			Synthetic (SubEM↑)		
	4K	8K	16K	4K	8K	16K	4K	8K	16K	4K	8K	16K
Full	5.00	4.85	4.73	48.00	55.20	68.60	56.75	55.25	56.50	98.31	97.44	89.62
P=2	5.17	5.04	4.92	33.40	50.20	63.00	55.75	53.00	51.25	89.50	87.19	85.50
P=4	5.43	5.33	5.26	19.20	38.20	57.00	50.50	47.25	44.50	64.69	71.19	65.69
P=8	5.88	5.84	5.83	11.00	23.00	44.20	44.00	40.75	39.00	16.06	16.31	18.31
P=16	6.70	6.97	7.35	6.80	10.60	22.00	37.75	34.00	30.00	1.31	0.75	0.56
P=32	8.55	10.09	11.96	6.00	6.60	8.00	34.00	20.75	14.25	0.38	0.06	0.12
P=64	11.45	15.24	20.65	3.60	4.20	6.40	34.00	8.50	2.25	0.00	0.00	0.00

Table 3: Comparisons between full-attention and naive parallel encoding with MISTRAL-7B-V0.3. Notations are the same as those in Table 1.

	LM (PPL↓)			ICL (Acc↑)			RAG (SubEM↑)			Synthetic (SubEM↑)		
	4K	8K	16K	4K	8K	16K	4K	8K	16K	4K	8K	16K
Full	7.33	7.27	7.01	29.60	42.80	53.20	63.50	66.00	60.75	68.44	60.69	67.50
P=2	7.53	7.50	7.25	34.20	35.20	51.20	62.00	57.50	57.00	67.88	41.06	37.06
P=4	8.15	7.72	7.71	23.20	34.60	44.40	58.25	54.25	51.00	25.38	45.00	13.88
P=8	8.51	8.97	8.13	17.80	28.20	45.60	53.00	48.25	48.50	4.12	3.56	14.06
P=16	9.23	9.62	10.56	12.20	17.40	31.00	46.25	39.75	38.75	0.31	0.50	1.25
P=32	10.40	10.92	11.96	6.40	10.20	21.40	42.75	38.00	33.25	0.00	0.00	0.00
P=64	12.12	13.09	14.67	4.00	5.60	11.40	42.75	29.75	25.50	0.00	0.00	0.00

Table 4: Comparisons between full-attention and naive parallel encoding with QWEN2-7B. Notations are the same as those in Table 1.

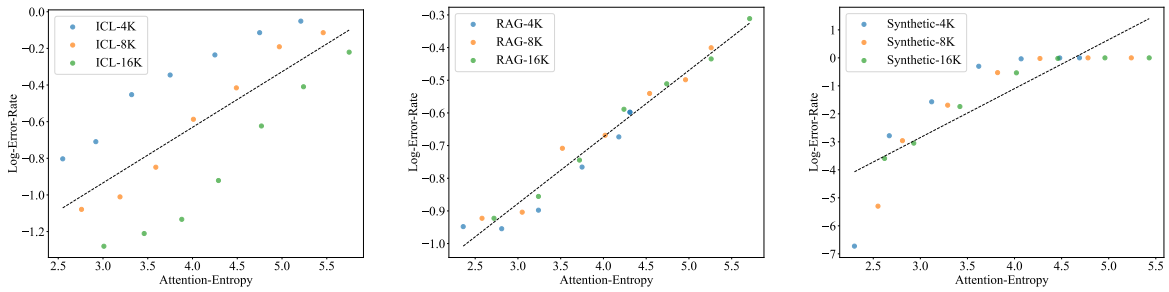

Figure 7: An illustration of the correlations between model performance and attention entropy on more tasks (with LLAMA-3.1-8B). Notations are similar to those in Figure 3.

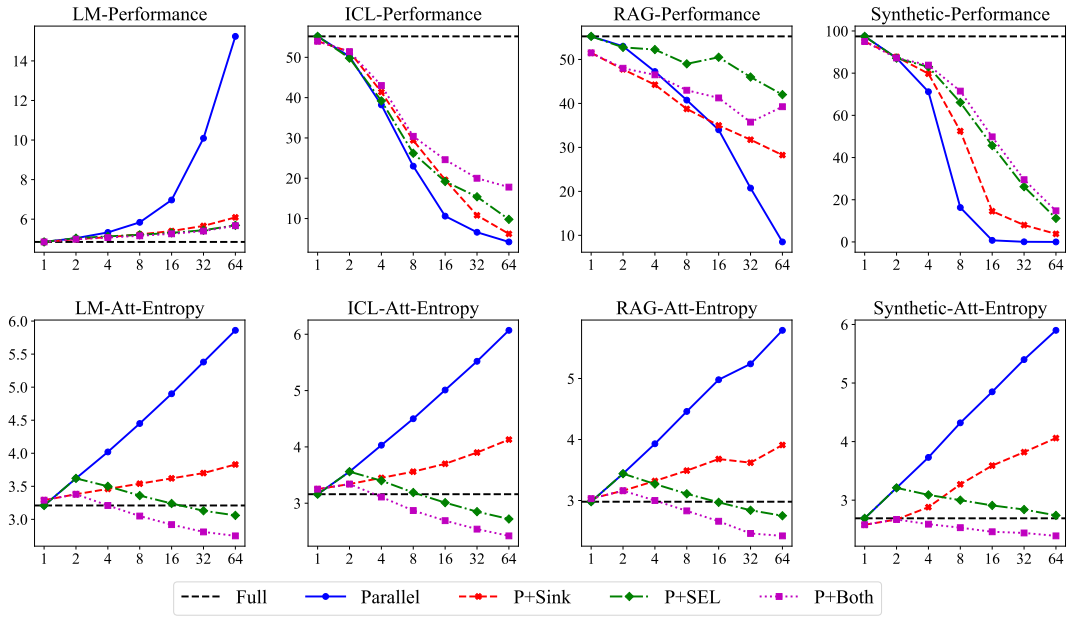

Figure 8: The influence of the entropy reduction methods (with MISTRAL-7B-V0.3 and 8K lengths). Notations are the same as those in Table 5.

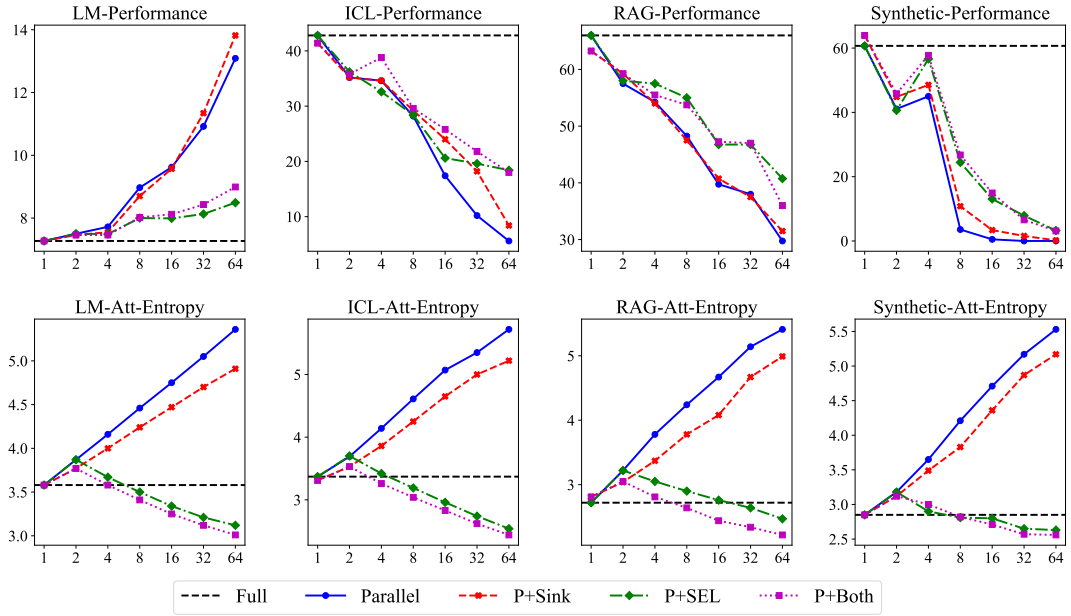

Figure 9: The influence of the entropy reduction methods (with QWEN2-7B and 8K lengths). Notations are the same as those in Table 5. Note that the "Sink" method seems to be less effective for Qwen, probably because it is less influenced by sink tokens, as evidenced by the less entropy reduction brought by "Sink".

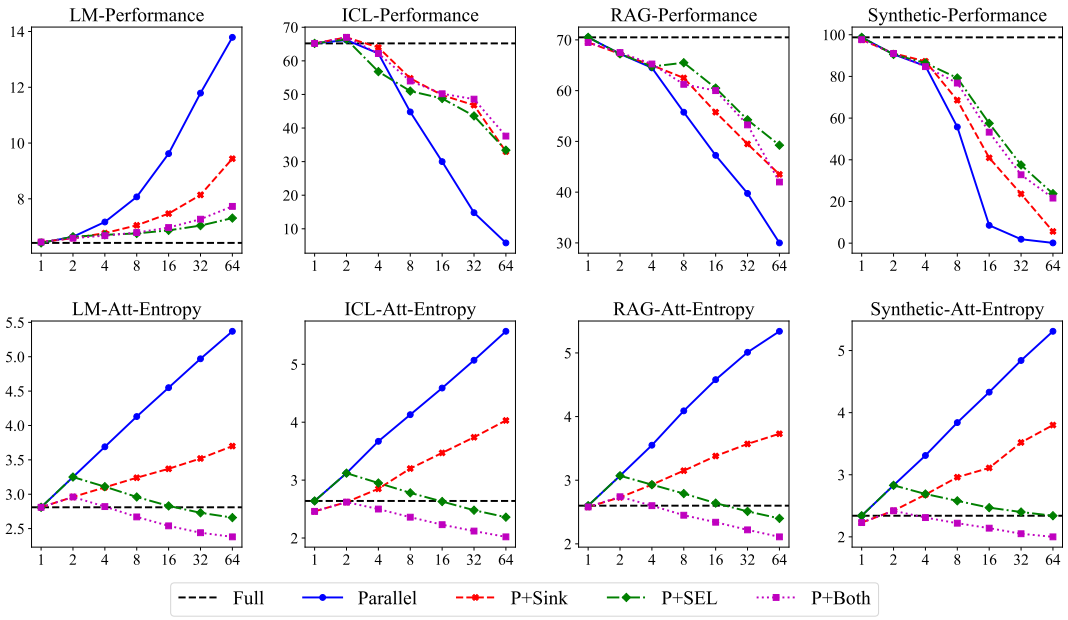

Figure 10: The influence of the entropy reduction methods (with LLAMA-3.1-8B-INSTRUCT and 8K lengths). Notations are the same as those in Table 5.

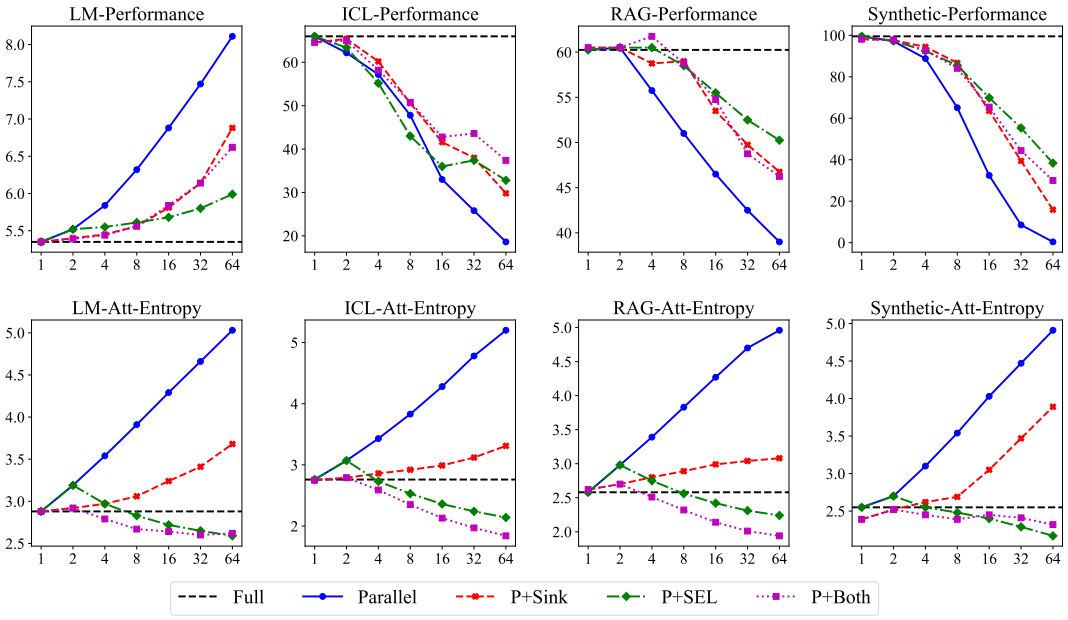

Figure 11: The influence of the entropy reduction methods using serialized position encoding (with LLAMA-3.1-8B and 8K lengths). Notations are the same as those in Table 5.