Reasoning-Focused Evaluation of Efficient Long-Context Inference Techniques

Joie Zhang[♠] **Qiyao Wei**[◇] **Howard Yen**[♠] **Xi Ye**[♠] **Danqi Chen**[♠]
[♠] Princeton Language and Intelligence [◇] University of Cambridge
joie@princeton.edu

Abstract

The rise of large reasoning models has demonstrated impressive capabilities in tackling increasingly complex tasks that often require generating long chains of thought. However, such capability also comes with a substantial increase in inference cost. Despite this growing trend, existing evaluations of efficient long-context inference techniques have primarily focused on preserving model performance for long inputs. We argue that this focus overlooks two crucial aspects of reasoning models: (1) their ability to generate long outputs, and (2) their ability to synthesize dispersed information – to reason over information distributed across both the input context and previously generated outputs. In this work, we systematically evaluate an array of efficient inference techniques across 12 tasks that differ in their requirements for long-output generation and information synthesis, on both instruction-tuned and reasoning models. We find that NF4 and Int8 weight quantization strongly outperform baselines and key-value (KV) cache token eviction methods on Pareto-optimality in both memory and performance across evaluated models, tasks, and context lengths. We observe that existing long-context efficient inference techniques perform especially poorly on tasks with long outputs and high token dispersion, which are critical for reasoning. Specifically, our experiments suggest that token eviction methods struggle in these settings because they cannot reliably perform exact string retrieval, likely because they tend to eject critical tokens during decoding. Finally, we find that using reasoning models can partially mitigate this degradation, recovering performance close to the full-cache baseline even at smaller cache sizes, thereby advancing the Pareto frontier of memory efficiency and long-context performance.

1 Introduction

Large Language Models (LLMs) that can process extensive context windows have unlocked new frontiers in complex reasoning [Bai et al., 2023]. In this landscape, reasoning models are particularly effective, as they articulate their process to deconstruct problems and use information distributed throughout a long context [Wei et al., 2022]. However, this advantage creates a practical tension: a model's verbose chain-of-thought (CoT), the source of its strength, can also cause context length explosion [Jaech et al., 2024]. This explosion directly inflates the size of the KV cache, which in turn increases peak memory usage, operational costs, and limits deployment at scale [Shazeer, 2019].

This paper investigates: can techniques like quantization and KV cache compression improve the Pareto efficiency (balancing performance and memory) for long-context reasoning tasks? How do different techniques, like quantization and token eviction, compare against each other? Finally, what are the failure modes of these techniques, and can reasoning models overcome them? To answer these questions, we evaluate 6 efficient inference techniques: weight quantization (NF4 and Int8) and token eviction for KV cache compression (DuoAttention, PyramidKV, SnapKV, and

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

StreamingLLM) [Dettmers et al., 2023, 2022, Xiao et al., 2024a, Li et al., 2024b, Xiao et al., 2024b]. These techniques are evaluated across 12 tasks that specifically test models' abilities in long-output generation and dispersed information synthesis [Ye et al., 2025, Goldman et al., 2024]. The contributions in our paper are as follows:

- We explore the performance-efficiency tradeoff among KV cache compressions methods. We benchmark methods by looking at the Pareto Frontier curve of performance (measured in accuracy) and efficiency (measured in peak memory usage). We find that both NF4 and Int8 quantization strongly outperform baseline models and token eviction methods on Pareto efficiency across both reasoning and instruct models.
- We evaluate not only standard long-input tasks but also those with long outputs and high token dispersion, settings that are crucial for reasoning. We empirically observe that current methods struggle in this regime.
- We find that the performance degradation of token eviction techniques are fundamentally due to brittleness on tasks that depend on exact string retrieval. Additionally, we demonstrate that reasoning models show promise in recovering this performance loss, achieving full performance even with smaller cache sizes and thereby advancing the Pareto frontier.

2 Evaluation Setup

2.1 Tasks

Past long-context evaluations of efficient inference techniques only consider the dimension of input context length, but we provide more comprehensive analysis by considering two additional dimensions: information dispersion and output generation length [Goldman et al., 2024].

We evaluate on 9 tasks from HELMET at the 16K and 32K context lengths and 3 tasks from LongProc at the 0.5K and 2K output lengths [Yen et al., 2025, Ye et al., 2025]. From HELMET, we choose a range of tasks that cover recall (NIAH and JSON KV), retrieval (HotpotQA and Natural Questions), citation generation (ASQA Cite), document reranking (MS MARCO), in-context learning (BANKING77 and CLINC150), and summarization (Multi-LexSum) because they encapsulate baselines like needle-in-a-haystack while also testing challenging reasoning tasks with high information dispersion [Kamradt, 2023, Yang et al., 2018, Liu et al., 2023, Kwiatkowski et al., 2019, Stelmakh et al., 2022, Gao et al., 2023, Bajaj et al., 2018, Casanueva et al., 2020, Larson et al., 2019]. From LongProc, we focus on tasks such as HTML-to-TSV, Pseudocode-to-Code, and Travel Planning, for assessing dispersed retrieval, procedural generation, and long-range reasoning capabilities [Li et al., 2024a, Kulal et al., 2019, Zheng et al., 2024]. Table 1 summarizes the dispersion and output length breakdown for each task.

Table 1: Task Categorization by Output Length and Dispersion

Output	Dispersion	Tasks	Description
Short	Low	NIAH	Retrieving a number within essays
		Recall JSON KV	Retrieve a key in JSON dictionary
		RAG HotpotQA	Multi-hop question answering
		RAG Natural Questions	Factoid question answering
Short	High	Cite	Answer ambiguous questions with citations
		Rerank	Rerank passage for a query
		ICL BANKING77	Banking intent classification, 77 labels
		ICL CLINC150	Intent classification, 151 labels
		Multi-LexSum	Summarizing multiple legal documents
Long	High	Pseudocode to Code	Translate pseudocode into C++ code
		HTML to TSV	Extract info from HTML into TSV
		Travel Planning	Craft a trip plan based on constraints

2.2 Efficient Inference Techniques

We analyze both input-independent and input-dependent methods to provide more robust coverage of reasoning and instruct model performance on efficient inference techniques. We choose Int8 and 4-bit NormalFloat (NF4) as our input-independent weight quantization techniques because they are leading standards for 8-bit and 4-bit quantization [Dettmers et al., 2022]. NF4 is an information-theoretically optimal quantization data type for normally distributed weights and empirically yields better results than 4-bit Integers and 4-bit Floats [Dettmers et al., 2023]. Additionally, for input-dependent methods, we evaluate on DuoAttention, which is the current state-of-the-art token eviction method, as well as SnapKV, PyramidKV, and StreamingLLM, to provide comprehensive coverage of the most canonical token eviction techniques [Xiao et al., 2024a, Li et al., 2024b, Cai et al., 2024, Xiao et al., 2024b].

2.3 Models

We choose to evaluate on the 8B parameter scale due to academic compute constraints and also because we aim to investigate the performance of smaller reasoning models on long-context reasoning tasks. We evaluate on Llama-3.1-8B-Instruct and Qwen2.5-7B-Instruct as our non-reasoning instruct models, which both have context lengths of 128K [Grattafiori et al., 2024, Team, 2024]. Additionally, we evaluate on DeepSeek-R1-Distill-Llama-8B and DeepSeek-R1-Distill-Qwen-7B for our reasoning models [Guo et al., 2025]. R1-Distill-Llama is distilled from the non-instruction-tuned base model of Llama-3.1-8B, which has a 128K context length, and R1-Distill-Qwen is distilled from Qwen2.5-Math-7B, which has an original maximum context window of 4K [DeepSeek-AI, 2025]. We report results from the R1-Distill models because they are distilled from the same model families and parameter scales as our baseline models, which helps to isolate the effect of reasoning on memory and performance.

3 Experiment Results

In this section, we discuss our experiment results and aim to investigate how different techniques, like quantization and token eviction, compare against each other in terms of Pareto efficiency (balancing performance and memory).

Our experimental results will be delivered in the following order: (1) Section 3.1 explores which efficient inference method has the overall strongest performance, aggregated across all evaluated tasks; (2) Section 3.2 analyzes how performance differs on tasks based on degree of information dispersion and output length; (3) Section 3.3 takes a closer look at performance on a task-by-task level; (4) Section 3.4 determines correlations between tasks; and (5) Section 3.5 and Section 3.6 contains case studies for qualitative and quantitative analysis of individual experiments.

3.1 Comparing Different Techniques

In Figure 1, we compare the average performance and peak memory usage of the baseline and six different methods on four models, averaged across all tasks from both HELMET (16K context) and LongProc (2K context) benchmarks. NF4 and Int8 achieve substantial memory savings of -48.46% ± 0.91% and -23.49% ± 3.73% respectively, with minimal performance degradation of only -2.46% ± 5.99% and -4.08% ± 6.13%. DuoAttention achieves near-baseline performance (-0.71% ± 2.82%) with minimal memory savings (-1.92% ± 0.04%). In contrast, token eviction methods (SnapKV, PyramidKV, StreamingLLM), averaged across multiple cache configurations, show substantial memory overhead (+29.53% ± 12.08%) with severe performance degradation (-22.85% ± 13.80%). When examining individual cache sizes for SnapKV and PyramidKV, we observe that smaller caches (w256, c2048) result in modest memory overhead (+8.45% ± 0.99%) but severe performance degradation (-24.92% ± 16.83%), while larger caches (w2048, c8192) substantially increase memory consumption (+67.47% ± 3.44%) and improve performance to -6.94% ± 8.83%. Specifically, increasing cache size from 2048 to 8192 improves performance by +32.93%, but requires +54.49% more memory. Thus, token eviction methods fail to achieve Pareto optimality: small caches sacrifice too much performance, while large caches consume substantially more memory than baseline while still underperforming it.

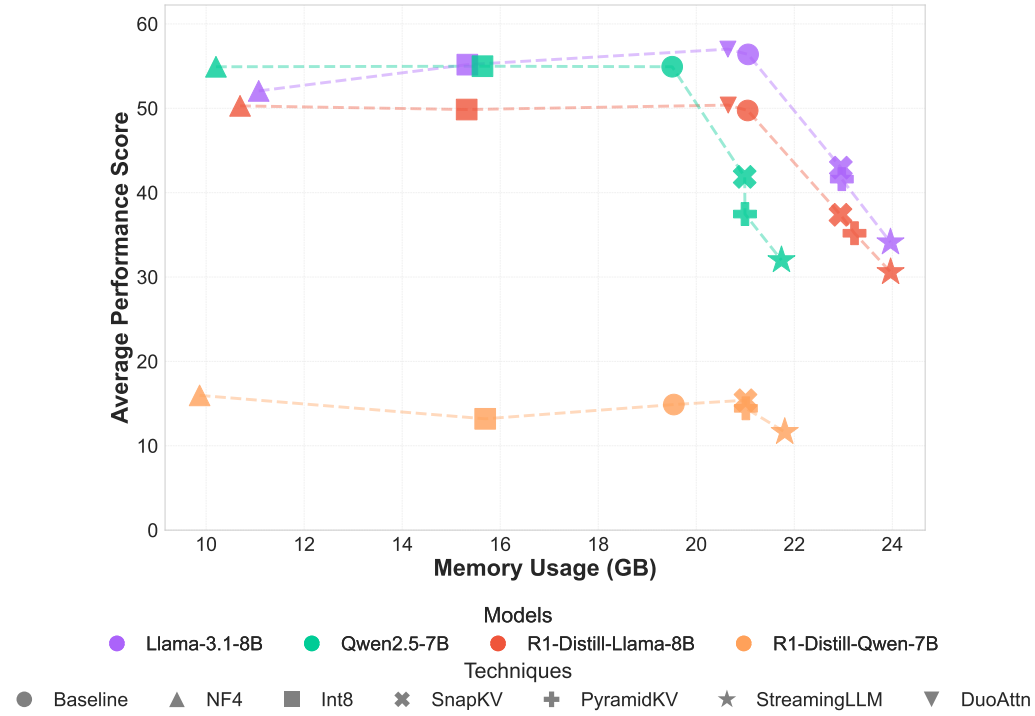

Figure 1: **Weight quantization is more Pareto-optimal than baseline and token eviction.** Performance–memory tradeoffs are presented for Llama-3.1-8B-Instruct, Qwen2.5-7B-Instruct, DeepSeek-R1-Distill-Llama-8B, and DeepSeek-R1-Distill-Qwen-7B on 6 efficient inference techniques and baseline. Results are averaged across all 12 tasks evaluated in this paper (9 HELMET tasks at 16K and 32K input and 3 LongProc tasks at 0.5K and 2K output). Note that DuoAttention results are limited to Llama-3.1-8B-Instruct and DeepSeek-R1-Distill-Llama-8B at 16K.

3.2 Comparing Different Tasks

In Figure 2, we examine the average performance of baseline and six different methods across three difficulty levels: Easy (Short Output, Low Dispersion), Medium (Short Output, High Dispersion), and Hard (Long Output, High Dispersion). The baseline model exhibits natural performance degradation as task difficulty increases, declining 45.3% from Easy to Medium tasks and an additional 23.9% from Medium to Hard tasks, representing an overall 58.4% drop across the difficulty spectrum. NF4 and Int8 demonstrate minimal performance degradation of -1.69% ± 2.18% and +0.48% ± 1.41% respectively, maintaining near-baseline performance across all difficulty levels. DuoAttention represents a notable exception among KV cache methods, achieving improved performance of +10.10% ± 4.74% across all difficulty levels, with particularly strong gains on Easy tasks (+18.6%) that diminish but remain positive on Medium (+9.4%) and Hard (+2.3%) tasks. In contrast, token eviction methods (SnapKV, PyramidKV, StreamingLLM) show substantial performance degradation averaging -25.62% ± 3.56% across all difficulty levels. StreamingLLM exhibits the most severe degradation at -34.14% ± 7.33%, particularly struggling on Easy tasks (-48.5%), while SnapKV and PyramidKV show more uniform degradation patterns at -19.24% ± 4.47% and -23.49% ± 4.38% respectively. Notably, token eviction methods exhibit substantial internal degradation as difficulty increases, with SnapKV and PyramidKV dropping 63.1% from Easy to Hard tasks, exacerbating the 58.4% baseline degradation. Thus, quantization methods maintain near-baseline performance with minimal variation across difficulty levels, DuoAttention consistently improves performance, while token eviction methods fail to achieve competitive performance across the difficulty spectrum.

We observe that DuoAttention performs exceptionally well on passkey-style long-input tasks, which happens to be the most common failure mode of other token eviction methods. We hypothesize this behavior occurs because the DuoAttention method identifies "retrieval heads" based on the attention heads that significantly alter model outputs when restricted to recent tokens and attention sinks from

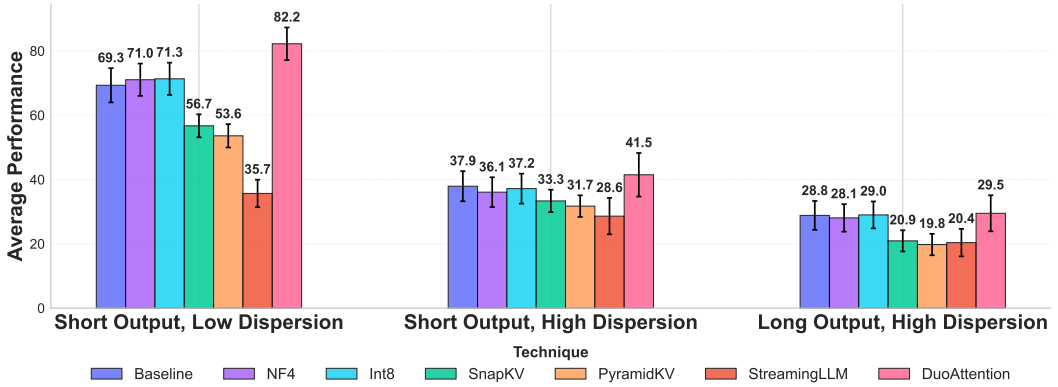

Figure 2: **Performance varies by task difficulty.** We categorize task difficulty along the dimensions of output length and dispersion: Easy (Short Output, Low Dispersion), Medium (Short Output, High Dispersion), Hard (Long Output, High Dispersion). Consistently, we find that quantization (NF4, Int8) outperforms token eviction methods across all difficulties, with the exception of DuoAttention. Notably, quantization methods exhibit minimal degradation relative to the baseline, maintaining comparable performance overall.

a passkey retrieval experiment [Xiao et al., 2024a]. In other words, DuoAttention faithfully produces the document IDs necessary for Re-rank because they are the type of passkey that DuoAttention's retrieval heads are designed to get right. However, it's also worth noting that DuoAttention is not perfect; it still experiences performance degradation compared to baseline on long-output tasks that require truly global context (Cite, Pseudocode to Code, HTML to TSV, and Travel Planning).

3.3 Task-by-Task Failure Mode Analysis

To better understand the failure modes of the evaluated methods, we examine in Figure 3 the performance degradation of each method relative to the baseline across all tasks. We find that token eviction methods exhibit the greatest performance degradation, particularly on the NIAH, Recall, and Re-rank tasks. These tasks share a common characteristic: they require precise string retrieval, both in low-dispersion settings (NIAH, Recall) and high-dispersion settings (Re-rank). This pattern suggests that the degradation of token eviction methods emerges from their inability to reliably perform exact string retrieval, likely because these methods tend to discard critical tokens during decoding. In one of our case studies in Section 3.5, we illustrate this phenomenon using the Re-rank task as an example. In this task, the model is required to rank documents by relevance, where each document ID consists of seven digits. However, due to token eviction, the model outputs only three digits, demonstrating how token eviction disrupts the model's ability to handle long-context tasks requiring precise string retrieval.

3.4 Calculating Correlations Between Tasks

To assess whether low-dispersion tasks can predict performance on high-dispersion and long-output tasks, we calculate the R^2 correlation on pairwise task performance for all model, context length, and efficient inference configurations. Figure 4 shows that synthetic, short-output low-dispersion tasks (NIAH, Recall, and RAG) may have some correlation with short-output high-dispersion tasks (ICL, Cite, Re-rank, and Summ) but low-dispersion tasks do not have strong correlation with long-output high-dispersion tasks (Pseudocode to Code, HTML to TSV, and Travel Planning) demonstrating the importance of evaluating along the axes of both output length and dispersion to capture a more comprehensive snapshot of long-context model performance.

3.5 Case Study: Why does token eviction perform poorly on ICL tasks? An analysis on Re-rank

To understand the performance degradation shown in Figure 2, we conduct a qualitative analysis of failure modes from the HELMET Re-rank task. We present here 4 representative examples where the

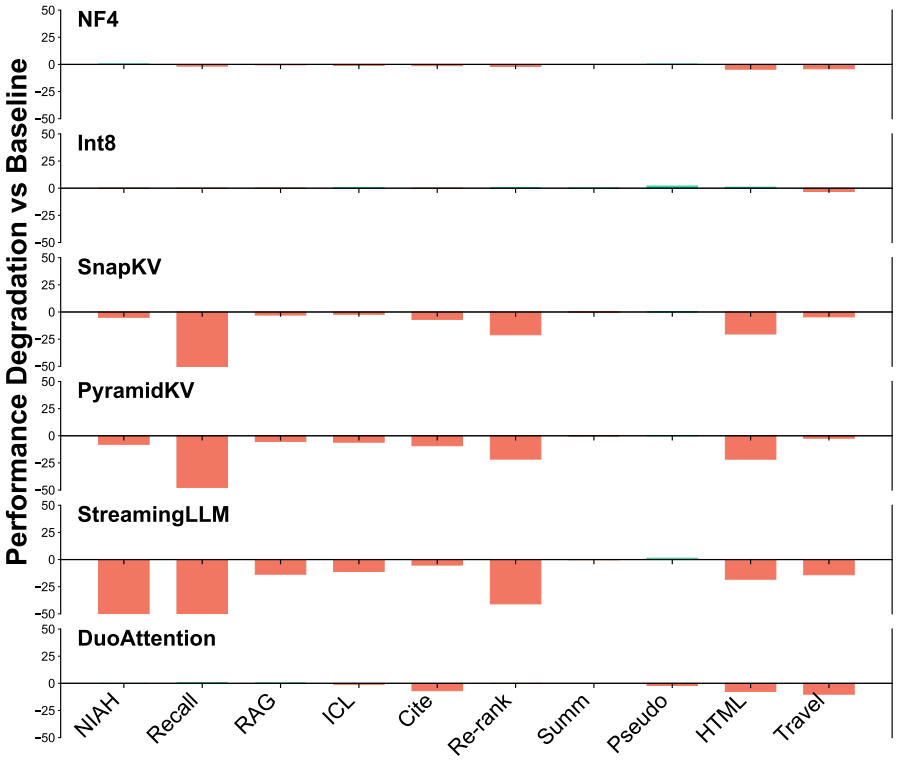

Figure 3: **Task-wise performance degradations relative to full-precision baseline.** Each subplot shows ΔPerformance (efficient technique minus baseline) for a different method, with positive values (green) indicating improvement and negative values (red) indicating degradation. DuoAttention shows minimal degradation on easy retrieval tasks but degrades on complex tasks. KV cache eviction methods (SnapKV, PyramidKV, StreamingLLM) exhibit severe degradation on recall and reranking tasks, with PyramidKV showing the largest drops. Quantization methods (NF4, Int8) maintain near-zero deltas across most tasks, preserving baseline performance. The performance deltas are the averaged results across models, context lengths, and cache configurations where applicable.

SnapKV token eviction method was used with the Llama-3.1-8B-Instruct model, with a window size of 32 tokens and a cache size of 256 tokens.

For the Re-rank task, the document IDs are typically 7 digits long. Across different queries, the model's output becomes decoupled from the semantic content of the query and documents. The model is not performing ranking based on relevance, instead generating arbitrary numerical patterns.

Moreover, sequentially decreasing and heavily repeating IDs suggest that SnapKV may have evicted in-context learning examples from its cache, thereby corrupting the model's internal representation of the documents in the context and returning the incorrect strings at test-time. By removing tokens that are critical for distinguishing between documents, the technique effectively blurs the available information. The model's outputs do not match the expected number of digits in the document IDs, thus failing the underlying task. These four failure modes demonstrate how token eviction methods are particularly sensitive to performance degradation on tasks that require perfect string retrieval.

3.6 Case Study: Reasoning Models are Better at Recovering Performance Degradation from Token Eviction Compared to Instruct Baselines on In-Context Learning Tasks

As discussed in Section 3.1, token eviction methods struggle more than baseline and quantization methods due to difficulty with in-context learning. What if reasoning models' strengths in in-context learning could be used to mitigate the performance degradation from token eviction methods?

Figure 6 shows the performance versus memory and latency of the 2 baseline models and 1 reasoning model on BANKING77 and CLINC150, two datasets that evaluate fine-grained intent classification.

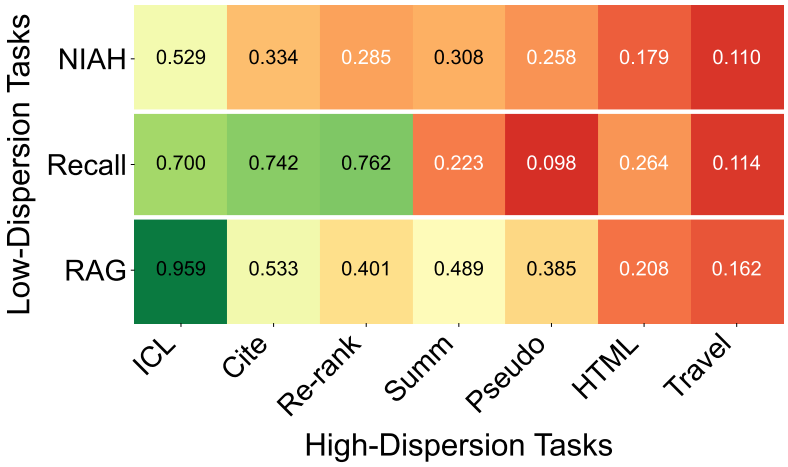

Figure 4: **Task-wise performance correlations across dispersion and output length.** We compute pairwise R^2 correlations between all tasks across model, context length, and efficient inference configurations to examine whether performance on low-dispersion tasks can predict that on more complex ones. Synthetic, short-output low-dispersion tasks (NIAH, Recall, RAG) show moderate correlation with short-output high-dispersion tasks (ICL, Cite, Re-rank, Summ). However, they exhibit weak or no correlation with long-output high-dispersion tasks (Pseudocode→Code, HTML→TSV, Travel Planning). These results highlight that evaluating only low-dispersion or short-output tasks provides an incomplete picture of long-context model performance, underscoring the need to assess along both dispersion and output-length axes.

Pattern 1: Sequential Numbers

Question: What types of food can you cook sous vide?

Answer ranking: 353 > 352 > 351 > 350 > 349 > ...

Pattern 3: Suspiciously Short IDs

Question: Causes of military suicide?

Answer ranking: 301 > 528 > 424 > 102 > 460 > ...

Pattern 2: Repeated ID

Question: Exons definition biology?

Answer ranking: 128 > 128 > 128 > 128 > 128 > ...

Pattern 4: Dominated by One ID

Question: Causes of military suicide?

Answer ranking: 220 > 152 > 220 > 220 > 220 > ...

Figure 5: **Qualitative Failure Modes of SnapKV on Re-rank.** Instead of returning 7-digit document IDs that correspond to the most relevant webpages that answer the user query, SnapKV is returning 3-digit IDs, suggesting that the critical tokens corresponding to the document IDs have been evicted from the cache. Moreover, the failure mode of sequentially decreasing and heavily repeating IDs suggests that SnapKV may have evicted in-context learning examples from its cache, thereby forgetting how to correctly complete the task.

Firstly, this ICL task is unique because it is one of the only three tasks where R1-Distill-Llama's baseline outperforms Llama-3.1-8B-Instruct's baseline. This finding suggests that perhaps, even at the 8B scale, reasoning models are able to take advantage of their long CoT planning capabilities to better learn from in-context examples.

Secondly, the most interesting finding is that the DeepSeek-R1-Distill-Llama-8B reasoning model demonstrates substantially greater resilience to token eviction techniques compared to the Llama-3.1-8B-Instruct model. Specifically, when subjected to aggressive token eviction with SnapKV at window size = 256 and cache size = 2048, Llama-Instruct experiences a 5.7 percentage point decrease (from 85.4% to 79.7%), while R1-Distill-Llama shows only a 0.8 percentage point decrease (from 87.1% to 86.3%). Similarly, with PyramidKV at the same cache configuration, Llama-Instruct drops 9.0 percentage points (from 85.4% to 76.4%), while R1-Distill-Llama drops only 2.2 percentage points (from 87.1% to 84.9%). The reasoning model thus exhibits 5.9 percentage points less degradation on

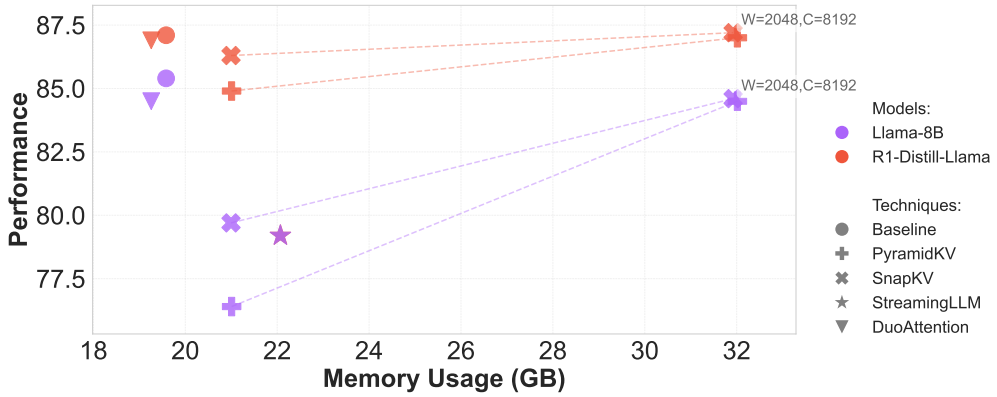

Figure 6: **Reasoning models mitigate performance degradation on ICL tasks.** Performance–memory tradeoffs are shown for one reasoning model and one non-reasoning model across five efficient inference techniques and a baseline on the ICL Banking and Clinc tasks. Baseline models exhibit moderate peak memory usage and low latency. Notably, DeepSeek-R1-Distill-Llama-8B (reasoning) mitigates performance degradation under token eviction methods, while Llama-3.1-8B-Instruct (non-reasoning) experiences greater drops at comparable cache sizes.

average than the instruct model even when subjected to the more ambitious token eviction setting. This superior resilience suggests that reasoning models may structure information more redundantly across their context, making them more robust to the loss of individual tokens during eviction.

In other words, reasoning models can help recover the performance degradation of token eviction methods for ICL tasks, which are precisely the tasks that token eviction methods struggle the most on. In this way, reasoning models can achieve nearly full performance on ICL even with smaller cache sizes, thus advancing the Pareto frontier. This finding is significant because model developers may be interested in using efficient inference techniques like token eviction to cut costs, but to mitigate the performance degradation of token eviction, they might consider using a small reasoning model for applications that require token eviction for less overall memory consumption.

4 Related Work

Long-Context Efficiency Techniques Efficient inference techniques for long-context models can be broadly divided into two categories: input-dependent and input-independent methods. Token eviction is a popular category of input-dependent methods that compress the KV cache via selective pruning and can be content-aware [Xiao et al., 2024b, Li et al., 2024b, Cai et al., 2024]. On the other hand, weight quantization is an input-independent method for optimizing the model's memory and latency, which means the performance gains are fixed and the method's performance is content-agnostic [Dettmers et al., 2022, 2023]. Positional interpolation is another input-independent method for extending the context windows of language models without incurring greater memory costs [Peng et al., 2024]. Existing evaluations are not assessing whether input-dependent or input-independent methods are more suitable for reasoning-intensive, long generation tasks for 8B-scaled models. This paper seeks to fill that gap, in addition to assessing how these different efficient inference methods interact with the long CoTs in reasoning models.

Long-Context Benchmarks While early studies of long-context models relied on synthetic tasks like "needle-in-a-haystack" retrieval, there has been an increased emphasis on real-world tasks involving multi-hop reasoning, long-input understanding, and long-output generation [Kamradt, 2023, Bai et al., 2023, Hsieh et al., 2024, Wu et al., 2025]. However, these datasets either focus primarily on retrieval without challenging reasoning or involve outputs too short to meaningfully test long-output generation. In this work, we evaluate on two benchmarks that represent more comprehensive tests of long-context reasoning capabilities. HELMET evaluates models on structured tasks including summarization, multi-hop QA, and logical reasoning over large inputs, thereby revealing retrieval and reasoning failures [Yen et al., 2025]. LongProc challenges models with complex, multi-step

operations over sequences of up to 8K in length, stress-testing both memory retention and the ability to maintain coherent chains-of-thought in long reasoning traces [Ye et al., 2025].

Efficient Inference in Reasoning Models Ever since the release of reasoning models, there has been strong interest in improving the efficiency of sampling and decrease the length of reasoning CoTs [Liu et al., 2025]. However, using KV cache compression for this task is largely overlooked. Song et al. [2025] explores this gap, but their paper only evaluates on AIME, which is reasoning-intensive but does not involve long-form generation, thus limiting the scope of the paper.

5 Conclusion

This work evaluates KV cache compression and quantization methods on long-input, long-output benchmarks, revealing critical performance gaps that are overlooked by simpler evaluations like NIAH. Our findings demonstrate that NF4 quantization consistently provides a Pareto-optimal balance of performance and memory efficiency, often surpassing more complex KV cache methods. We identify a key failure mode for token eviction strategies, which struggle significantly on tasks requiring precise recall of identifiers, as they are prone to discarding vital information. Furthermore, specialized methods like DuoAttention show limitations on tasks that do not align with their passkey-retrieval-focused fine-tuning.

Ultimately, our results underscore that the effectiveness of a long-context inference technique is highly task-dependent. Benchmarks that test both long-range information synthesis and extended generation, such as HELMET and LongProc, are essential for developing robust and practical LLMs. The widespread success on NIAH alongside struggles on these more complex tasks confirms that the field must move toward more realistic benchmarks to drive meaningful progress.

Looking ahead, several promising research directions emerge from this work. First, future efforts could focus on hybrid methods, combining the raw efficiency of quantization with more sophisticated, content-aware KV cache eviction strategies to potentially push the Pareto frontier further. Second, developing adaptive inference techniques that can dynamically adjust their cache management policy based on the inferred task type (e.g., retrieval-heavy vs. generation-heavy) could yield significant performance gains. Finally, extending this evaluation methodology to a wider range of models, including but not limited to reasoning models, would be crucial to understanding how these efficiency techniques interact with models with different capabilities.

6 Limitations

While our findings provide valuable insights into the trade-offs between reasoning models and efficient inference techniques, it is worth acknowledging our study's limitations.

First, our analysis is focused on models at the 7B and 8B parameter scale due to academic compute constraints. The dynamics of CoT verbosity, context length explosion, and the effectiveness of inference techniques may differ substantially in much larger models (e.g., 70B+), where reasoning capabilities are more pronounced. Consequently, our conclusions about the Pareto efficiency of smaller reasoning models may not generalize to their larger counterparts.

Second, the selection of tasks, while intentionally broad, is not exhaustive. Our key finding—that reasoning models can rescue the performance of token eviction—was most evident on in-context learning (ICL) tasks. This benefit did not extend to tasks like document reranking, which are highly sensitive to information loss. The trade-offs between model type and efficiency technique are highly task-dependent, and other long-context domains such as creative writing or complex code repository analysis may present these trade-offs differently.

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. LongBench: A Bilingual, Multitask Benchmark for Long Context Understanding. *arXiv preprint arXiv:2308.14508*, 2023.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. Ms marco: A human generated machine reading comprehension dataset, 2018. URL https://arxiv.org/abs/1611.09268.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020. URL https://arxiv.org/abs/2004.05150.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Baobao Chang, Junjie Hu, and Wen Xiao. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling, 2024. URL https://arxiv.org/abs/2406.02069.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders. In Tsung-Hsien Wen, Asli Celikyilmaz, Zhou Yu, Alexandros Papangelis, Mihail Eric, Anuj Kumar, Iñigo Casanueva, and Rushin Shah, editors, *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.nlp4convai-1.5. URL https://aclanthology.org/2020.nlp4convai-1.5.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35: 30318–30332, 2022.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6465–6488, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.398. URL https://aclanthology.org/2023.emnlp-main.398.

Omer Goldman, Alon Jacovi, Aviv Slobodkin, Aviya Maimon, Ido Dagan, and Reut Tsarfaty. Is It Really Long Context if All You Need Is Retrieval? Towards Genuinely Difficult Long Context NLP. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, November 2024. URL https://aclanthology.org/2024.emnlp-main.924.

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

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H Abdi, Dongsheng Li, Chin-Yew Lin, et al. Minference 1.0: Accelerating pre-filling for long-context llms via dynamic sparse attention. *Advances in Neural Information Processing Systems*, 37:52481–52515, 2024.

Gregory Kamradt. Needle In A Haystack - pressure testing LLMs, 2023. URL https://github.com/gkamradt/LLMTest_NeedleInAHaystack/tree/main.

Sumith Kulal, Panupong Pasupat, Kartik Chandra, Mina Lee, Oded Padon, Alex Aiken, and Percy S Liang. Spoc: Search-based pseudocode to code. *Advances in Neural Information Processing Systems*, 32, 2019.

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. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026.

Xunhao Lai, Jianqiao Lu, Yao Luo, Yiyuan Ma, and Xun Zhou. Flexprefill: A context-aware sparse attention mechanism for efficient long-sequence inference. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OfjIlbelrT.

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. An evaluation dataset for intent classification and out-of-scope prediction. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *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, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1131. URL https://aclanthology.org/D19-1131.

Xiang Li, Xiangyu Zhou, Rui Dong, Yihong Zhang, and Xinyu Wang. Efficient bottom-up synthesis for programs with local variables. *Proceedings of the ACM on Programming Languages*, 8(POPL): 1540–1568, 2024a.

Yucheng Li, Huiqiang Jiang, Qianhui Wu, Xufang Luo, Surin Ahn, Chengruidong Zhang, Amir H. Abdi, Dongsheng Li, Jianfeng Gao, Yuqing Yang, and Lili Qiu. Scbench: A kv cache-centric analysis of long-context methods, 2025. URL https://arxiv.org/abs/2412.10319.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. Snapkv: Llm knows what you are looking for before generation, 2024b. URL https://arxiv.org/abs/2404.14469.

Di Liu, Meng Chen, Baotong Lu, Huiqiang Jiang, Zhenhua Han, Qianxi Zhang, Qi Chen, Chengruidong Zhang, Bailu Ding, Kai Zhang, Chen Chen, Fan Yang, Yuqing Yang, and Lili Qiu. Retrievalattention: Accelerating long-context llm inference via vector retrieval, 2024a. URL https://arxiv.org/abs/2409.10516.

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

Xiang Liu, Peijie Dong, Xuming Hu, and Xiaowen Chu. LongGenBench: Long-context generation benchmark. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 865–883, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.48. URL https://aclanthology.org/2024.findings-emnlp.48/.

Yue Liu, Jiaying Wu, Yufei He, Ruihan Gong, Jun Xia, Liang Li, Hongcheng Gao, Hongyu Chen, Baolong Bi, Jiaheng Zhang, Zhiqi Huang, Bryan Hooi, Stan Z. Li, and Keqin Li. Efficient inference for large reasoning models: A survey, 2025. URL https://arxiv.org/abs/2503.23077.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient context window extension of large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

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

Noam Shazeer. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*, 2019.

Jiwon Song, Dongwon Jo, Yulhwa Kim, and Jae-Joon Kim. Reasoning path compression: Compressing generation trajectories for efficient llm reasoning, 2025. URL https://arxiv.org/abs/2505.13866.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. ASQA: Factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8273–8288, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.566. URL https://aclanthology.org/2022.emnlp-main.566.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference, 2024. URL https://arxiv.org/abs/2406.10774.

Qwen Team. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2:3, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903, 2022.

Yuhao Wu, Ming Shan Hee, Zhiqiang Hu, and Roy Ka-Wei Lee. Longgenbench: Benchmarking long-form generation in long context LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=3A71qNKWAS.

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

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks, 2024b. URL https://arxiv.org/abs/2309.17453.

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

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259.

Jiayi Yao, Hanchen Li, Yuhan Liu, Siddhant Ray, Yihua Cheng, Qizheng Zhang, Kuntai Du, Shan Lu, and Junchen Jiang. Cacheblend: Fast large language model serving for rag with cached knowledge fusion, 2025. URL https://arxiv.org/abs/2405.16444.

Xi Ye, Fangcong Yin, Yinghui He, Joie Zhang, Howard Yen, Tianyu Gao, Greg Durrett, and Danqi Chen. Longproc: Benchmarking long-context language models on long procedural generation. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=ruWC5LIMSo.

Howard Yen, Tianyu Gao, Minmin Hou, Ke Ding, Daniel Fleischer, Peter Izsak, Moshe Wasserblat, and Danqi Chen. Helmet: How to evaluate long-context language models effectively and thoroughly. In *International Conference on Learning Representations (ICLR)*, 2025.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. ∞bench: Extending long context evaluation beyond 100k tokens, 2024. URL https://arxiv.org/abs/2402.13718.

Huaixiu Steven Zheng, Swaroop Mishra, Hugh Zhang, Xinyun Chen, Minmin Chen, Azade Nova, Le Hou, Heng-Tze Cheng, Quoc V Le, Ed H Chi, et al. Natural plan: Benchmarking llms on natural language planning. *arXiv preprint arXiv:2406.04520*, 2024.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi : Plug-and-play 2bit kv cache quantization with streaming asymmetric quantization. 2023. doi: 10.13140/RG.2.2.28167.37282. URL https://rgdoi.net/10.13140/RG.2.2.28167.37282.

A Extended Related Work

A.1 Long-Context Efficiency Techniques

Efficient inference techniques for long-context models can be broadly divided into two categories: direct KV cache optimizations and auxiliary scaling techniques. KV cache techniques primarily alter memory handling by restructuring, compressing, retrieving, or dynamically loading key-value pairs during inference, while auxiliary techniques target scaling efficiency more broadly without directly modifying the KV cache.

Recent research has identified four primary strategies for KV cache optimization: [Jiang et al., 2024]

1. **KV Cache Generation**: Techniques that proactively design the structure of the KV cache to reduce memory or computational demands during prefill and decoding.
   Examples: MInference [Jiang et al., 2024], FlexPrefill [Lai et al., 2025], A-shape [Xiao et al., 2024b], Tri-shape [Li et al., 2025], Dilated Attention [Beltagy et al., 2020].
2. **KV Cache Compression**: Techniques that selectively prune, merge, or compress KV entries during decoding to maintain a manageable memory footprint even as context grows.
   Examples: StreamingLLM [Xiao et al., 2024b], SnapKV [Li et al., 2024b], PyramidKV [Cai et al., 2024], KIVI [Zirui Liu et al., 2023].
3. **KV Cache Retrieval**: Techniques that store large external KV memories and dynamically retrieve only the most relevant tokens when needed, reducing active memory load.
   Examples: CacheBlend [Yao et al., 2025].
4. **KV Cache Loading**: Techniques that manage offloading and re-loading of KV cache chunks (e.g., from disk or remote storage) during decoding, enabling arbitrarily long contexts without linear memory growth.
   Examples: Quest [Tang et al., 2024], RetrievalAttention [Liu et al., 2024a].

A.2 Auxiliary Efficiency Techniques

In addition to direct KV cache optimizations, several auxiliary techniques have emerged to support scaling to longer contexts and reducing inference costs, even though they are not specifically designed for KV cache handling.

One important auxiliary technique is **positional interpolation**, exemplified by methods like YaRN [Peng et al., 2024]. Positional interpolation modifies the way positional encodings are calculated for input tokens. Rather than using absolute or fixed positional embeddings, interpolation allows the model to flexibly re-map positions across very long sequences. This enables models trained on shorter contexts to extrapolate to longer ones without significant degradation in performance.

Another critical auxiliary technique is **quantization**, which reduces the memory footprint and computational cost by decreasing the precision of model activations and weights. In standard training and inference, models often operate using **bf16** (Brain Floating-Point 16) precision, which uses 16 bits to represent floating-point numbers with high dynamic range and some precision trade-offs.

By contrast, **int8** and **int4** quantization compress model weights and activations into 8-bit or 4-bit integers, respectively. This drastic reduction in precision significantly decreases both memory bandwidth and storage requirements, enabling faster and more efficient inference, especially important for resource-constrained deployment. However, aggressive quantization (like int4) can introduce noticeable performance degradation if not handled carefully with quantization-aware techniques.

Together, positional interpolation and quantization serve as complementary tools to core KV cache optimizations, broadening the model's ability to scale efficiently to longer inputs while mitigating hardware and memory bottlenecks.

A.3 Long-Context Benchmarking Techniques

Despite significant advances in efficient inference methods, many recent papers continue to rely heavily on perplexity as the primary evaluation metric. While perplexity offers a coarse measure of language modeling ability, it often fails to capture failures in long-range reasoning, information retention, and multi-step computation that are critical for practical applications of long-context

models. Especially in the context of techniques like KV cache compression or retrieval-augmented inference, perplexity can mask deeper degradation in reasoning chains or factual consistency, leading to an incomplete assessment of model robustness.

While early studies of long-context models relied on synthetic tasks—such as "needle-in-a-haystack" retrieval—there has been increasing recognition that real-world tasks demand deeper multi-hop reasoning, long input understanding, and long output generation. Consequently, a range of benchmarks have emerged to stress-test these abilities, ZeroSCROLLS [Shaham et al., 2023], LongBench [Bai et al., 2023], L-Eval [Yang et al., 2024], RULER [Hsieh et al., 2024], ∞ BENCH [Zhang et al., 2024], LongGenBench1 [Wu et al., 2025], and LongGenBench2 [Liu et al., 2024b].

However, not all benchmarks equally capture the complex interaction between memory and reasoning. Many datasets either focus primarily on retrieval (without challenging reasoning chains) or involve outputs that are too short to meaningfully test long-output generation.

In this work, we focus on two benchmarks that represent some of the most comprehensive tests for long-context capabilities:

- **LongProc**: Focused on procedural reasoning, LongProc challenges models to perform complex, multi-step operations over very long input sequences. This benchmark stresses not only memory retention but also the ability to maintain coherent logical chains while processing and generating thousands of tokens.
- **HELMET**: Designed to systematically probe long-context abilities across a wide range of tasks, HELMET evaluates models on structured tasks involving summarization, multi-hop QA, and logical reasoning over large textual inputs. Its diverse task types and careful construction make it an ideal choice for diagnosing both retrieval and reasoning failures.

Together, LongProc and HELMET provide a robust evaluation landscape for our study. By relying on these benchmarks, we can rigorously test how well efficient inference techniques preserve multi-hop reasoning ability and context fidelity across both long-input and long-output use cases.

B Additional Experimental Results

B.1 Hyperparameter Selection for PyramidKV.

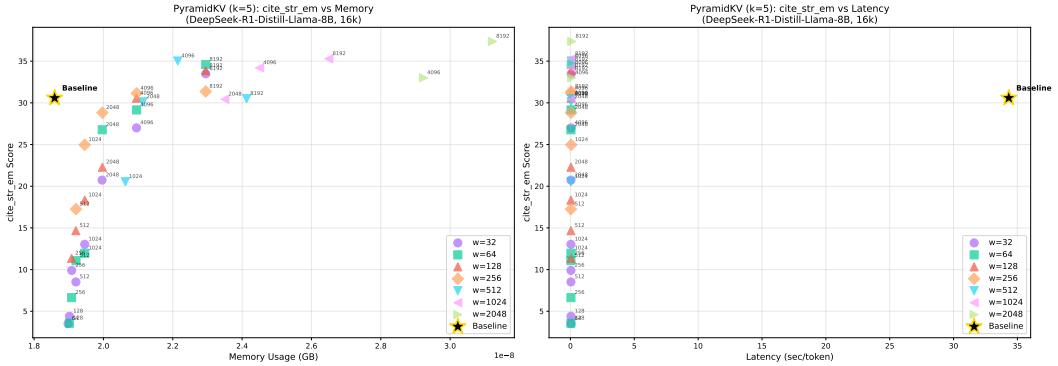

Figure 7: **Hyperparameter sweep for PyramidKV.** In order to pick the optimal hyperparameters for PyramidKV, we ran a sweep over the window size and KV cache size given kernel size equals 5 using the Cite String Exact Match metric. We find that increasing the cache size too much yields diminishing returns in performance while increasing memory consumption. We also observe that a window size equals 256 gives the optimal curve on the Pareto frontier.

B.2 Full Pair-wise Task Correlations

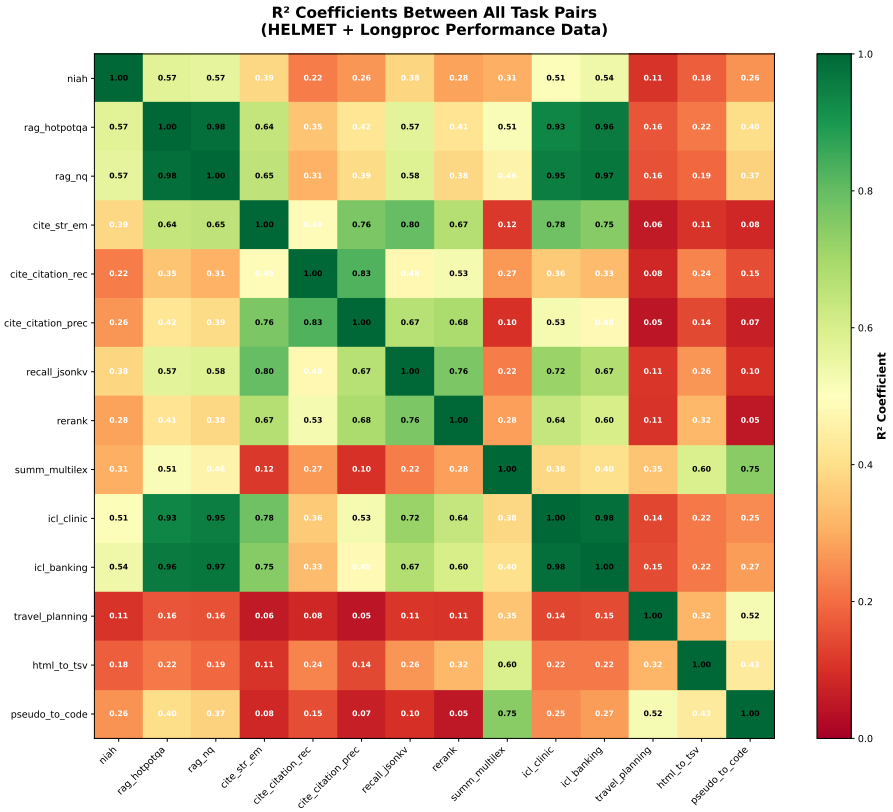

Figure 8: Comprehensive pairwise correlations between all evaluated tasks. Each cell represents the coefficient of determination (R^2) between task performance scores, capturing how strongly performance on one task predicts performance on another. Higher values indicate stronger alignment in method behavior across tasks, while lower values highlight task-specific divergences.