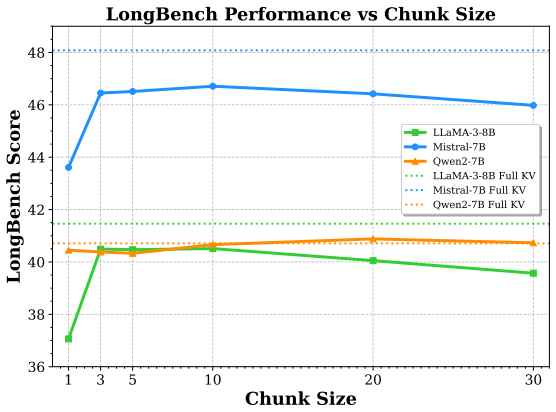

Figure 5: LongBench Performance Comparison with different chunk size under 10% compression rate.

on index reuse, please refer to the APPENDIX B.1.3.

Overall, these findings on efficiency and performance suggest that layer-wise index reuse can be an effective technique for optimizing the efficiency-performance trade-off in KV cache compression, with the potential for model-specific tuning to maximize benefits.

5. Ablation study

5.1. Chunk Size

This section aims to investigate the impact of chunk size on the performance of ChunkKV. Different chunk sizes will lead to varying degrees of compression on the semantic information of the data. We set the experiemnt setting the same as in LongBench in Section 4.2. The chunk size is set from the range $\{1, 3, 5, 10, 20, 30\}$. Figure 5 shows the performance of the ChunkKV with different chunk size on the LongBench and NIAH benchmarks. The three colorful curves represent three LLMs with different chunk sizes, and the colorful dashed line is the corresponding FullKV performance. For more experiments on the size of the chunks with different compression ratios, refer to the Appendix B.4.

Table 9: LongBench Performance with Different Chunk Sizes and Compression Ratios for LLaMA-3-8B-Instruct

Compression	Chunk Size						
Rate	1	3	5	10	15	20	30
10%	37.32	40.49	40.47	**40.51**	40.21	40.05	39.57
20%	38.80	40.66	40.57	**40.74**	40.53	40.46	40.04
30%	39.23	41.02	41.29	**41.59**	41.38	41.33	41.02

From Figure 5, we can observe that the LongBench performance of ChunkKV is not significantly affected by the chunk size, with performance variations less than 1%. The three curves are closely aligned, indicating that chunk sizes in the range of $\{10, 20\}$ exhibit better performance.

Table 9 and 10 show the performance of ChunkKV with different comperession ratios and different chunk sizes on the LongBench and NIAH. We conducted extensive experiments across different compression ratios and KV cache sizes to shows the effectiveness of ChunkKV and the chunk size is robust.

Table 10: NIAH Performance with Different Chunk Sizes and KV Cache Sizes for LLaMA-3-8B-Instruct

KV Cache	Chunk Size						
Size	1	3	5	10	15	20	30
96	41.0	63.2	65.2	**70.3**	67.2	65.3	53.1
128	47.9	65.6	69.1	**73.8**	72.3	72.0	71.2
256	61.7	70.3	71.2	**74.1**	73.2	72.3	71.1
512	68.6	72.6	72.5	**74.5**	74.3	74.0	72.6

From the chunk size ablation study, we can observe that across different tasks (LongBench and NIAH) and various compression settings, a chunk size of 10 consistently delivers optimal or near-optimal performance. This empirical finding suggests that a chunk size of 10 strikes a good balance between preserving semantic information and compression efficiency, making it a robust default choice for ChunkKV. Therefore, we adopt this chunk size setting throughout our experiments.

6. Conclusion

We introduced ChunkKV, a novel KV cache compression method that preserves semantic information by retaining more informative chunks. Through extensive experiments across multiple state-of-the-art LLMs (including DeepSeek-R1, LLaMA-3, Qwen2, and Mistral) and diverse benchmarks (GSM8K, LongBench, NIAH, and JailbreakV), we demonstrate that ChunkKV consistently outperforms existing methods while using only a fraction of the memory. Our comprehensive analysis shows that ChunkKV's chunk-based approach maintains crucial contextual information, leading to superior performance in complex reasoning tasks, long-context understanding, and safety evaluations. The method's effectiveness is particularly evident in challenging scenarios like many-shot GSM8K and multi-document QA tasks, where semantic coherence is crucial. Furthermore, our proposed layer-wise index reuse technique provides significant computational efficiency gains with minimal performance impact, achieving up to 20.7% latency reduction and 26.5% throughput improvement. These findings, supported by detailed quantitative analysis and ablation studies, establish ChunkKV as a significant advancement in KV cache compression technology, offering an effective solution for deploying LLMs in resource-constrained environments while maintaining high-quality outputs.