SGD-KV: Summarization Guided KV Cache Compression

Zeyu Liu* USC, USA
Woomin Song KAIST, Korea
Xuandi Fu Amazon AGI, USA
Sai Muralidhar Jayanthi Amazon AGI, USA

Vivek Govindan Amazon AGI, USA
Aram Galstyan Amazon AGI, USA
Sravan Babu Bodapati Amazon AGI, USA
Srikanth Ronanki Amazon AGI, USA

Abstract

Large language models (LLMs) face severe memory bottlenecks in long-context inference due to the linearly growing size of key-value (KV) caches. Existing KV cache compression techniques typically rely on simple heuristics, overlooking the distinct functional roles of different attention heads. We present SGD-KV (Summarization-Guided KV Cache Compression), a head-aware framework that leverages a novel chunk-summarization diagnostic task to systematically identify and prioritize attention heads specialized in hierarchical information aggregation. Experiments on Qwen2.5-7B-1M and Qwen3-32B across diverse long-context benchmarks demonstrate that SGD-KV achieves state-of-the-art performance with contexts up to 1M tokens, while reducing KV cache memory usage by up to 75%. Our findings show that strategically allocating the KV cache budget based on the summarization score distribution of attention heads yields a superior efficiency–accuracy trade-off for long-context inference.

1 Introduction

The push towards million-token context windows in Large Language Models (LLMs) like Qwen2.5-1M [1] and Gemini 2.5 [2] is severely hampered by a fundamental obstacle: the prohibitive memory cost of the Key-Value (KV) cache, which scales linearly with context length. While methods exist to compress the KV cache, from early token-level eviction to more recent head-aware approaches [3, 4], they often rely on simple, retrieval-based importance metrics to allocate cache budget. These heuristics fall short in complex scenarios like multi-document analysis or long-form dialogue, which demand hierarchical information aggregation rather than simple pattern matching. We argue that a specialized subset of attention heads, which we term "summarization heads," are primarily responsible for this higher-order cognitive function. To leverage this insight, we introduce SGD-KV (Summarization-Guided KV Cache compression), a framework that systematically identifies and prioritizes these crucial heads. Using a novel chunk-summarization diagnostic task, SGD-KV scores each head's summarization capability and applies a water-filling-inspired algorithm to intelligently allocate cache budget for each head. Validated on benchmarks including OpenAI Multi-Round Co-Reference Resolution (MRCR) [5] and ETHIC [6], SGD-KV sets a new state-of-the-art (SOTA), reducing KV cache usage by up to 75% on contexts up to 1 million tokens without compromising model accuracy.

2 Related Work

KV Cache Compression Research on mitigating the memory burden of KV caches has progressed along two fronts: token-level eviction and head-level budget allocation. Early efforts like

*Work done during an internship at Amazon. Correspondence to: Srikanth Ronanki <ronanks@amazon.com>

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

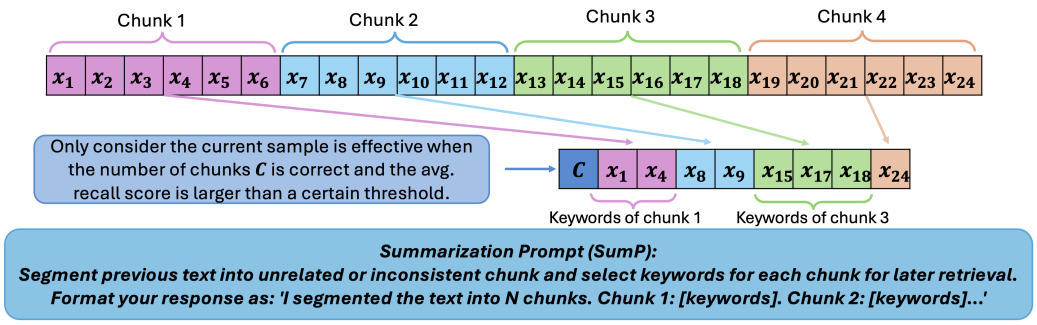

Figure 1: The illustration of chunk-summarization task.

StreamingLLM [7] and H2O [8] focused on identifying and discarding less important tokens based on heuristics like recency and cumulative attention scores. While reducing memory, these methods are functionally unaware. More sophisticated approaches like PyramidKV [9] and AdaKV [4] introduce dynamic cache allocation across different layers and heads, but their strategies are still guided by quantitative attention patterns rather than the semantic role of each head.

Attention Head Specialization Concurrently, research into attention mechanism interpretability has demonstrated that heads often specialize. This was first shown for "retrieval heads" identified via needle-in-a-haystack tasks [10], and later for other pattern-matching roles like induction heads [11] and retrieval-reasoning (R2) heads [3]. However, these functional discoveries have been driven by retrieval-centric diagnostics, overlooking heads specialized for higher-order cognitive tasks. Our work bridges this divide: we introduce a summarization-based task to identify heads that perform hierarchical information synthesis and, for the first time, leverage this functional insight to create a more intelligent and efficient KV cache compression strategy.

3 Method

3.1 Chunk-Summarization Task Design

To effectively measure a head's ability to aggregate information, we design a diagnostic task that requires multi-level understanding, going beyond simple pattern matching. As illustrated in Figure 1, we first construct long-context samples by concatenating multiple documents from various short-text summarization datasets (e.g., CNN/Dailymail dataset [12], DialogSum dataset [13]) and record the boundaries of each original document. Given a concatenated sample, we prompt the model to perform a two-part task: (1) **Chunk Identification**: Identify the number of distinct semantic chunks within the concatenated document; (2) **Keyword Extraction**: For each identified chunk, generate a concise list of keywords that capture its core meaning. This design compels the model to first parse the document structure at a high level and then distill the semantic essence of each segment, providing a strong signal for identifying heads involved in hierarchical aggregation.

3.2 Summarization Score Calculation

To quantify each head's summarization capability, we first filter for valid model responses as shown in Fig. 1 and then compute an attention-based importance score. For each valid sample, we compute an importance score I_h for every attention head h. The score measures the strength of attention from the generated keywords back to their corresponding source text within the correct chunk.

$$I_h = \frac{1}{c}\sum_{m=1}^{c}\frac{1}{|\mathcal{K}_m|}\sum_{j\in\mathcal{K}_m}\max_{n} A_h(p_j^o, p_{j,n}^i) \tag{1}$$

where $\mathcal{K}_m$ is the set of keywords for chunk m, p_j^o is the position of an output keyword token j, and $p_{j,n}^i$ is the n^{th} occurrence of that same keyword in the input text. The $\max$ operation identifies the strongest attention link from a generated keyword to its most salient source token, effectively capturing the head's ability to pinpoint and aggregate key information. The final summarization score for each head is its average importance score across all valid samples.

3.3 KV Cache Budget Allocation

The computed summarization scores directly guide our head-aware KV cache allocation strategy. Similar to HeadKV [3], we first normalize the scores across all heads such that they sum to one, then we always preserve a fixed budget for initial sink tokens (b_{sink}) and a sliding window of recent tokens (b_{window}) as the instructions of users are usually in the beginning or end of the whole input in practice.

The remaining cache budget for the middle, compressible portion of the sequence ($M = C - b_{\text{sink}} - b_{\text{window}}$, where C is total sequence length) is then allocated to each head h in proportion to its normalized summarization score S_h, at a given overall KV cache budget ratio R:

$$b_h = (S_h \cdot L \cdot H \cdot R) \cdot M + b_{\text{sink}} + b_{\text{window}} \tag{2}$$

where L is the number of layers, and H is the number of KV heads. This ensures that heads identified as crucial for summarization receive a larger cache budget, allowing them to retain more historical context. We further adapt the Water-filling algorithm to reallocate the excessive KV budget from heads with very high scores to other heads with comparatively high scores, as detailed in Appendix A.1. Within each head's allocated budget b_h, we use a standard token selection mechanism based on cumulative attention scores, similar to SnapKV [14], to select the most important KV entries to keep.

4 Experiments

Experimental Setup We evaluate our approach on Qwen2.5-7B-Instruct-1M [1] and Qwen3-32B [15] as representative models for non-reasoning small LMs and reasoning-capable large LMs, respectively. However, we observed that the Qwen2.5-7B-Instruct-1M checkpoint from HuggingFace exhibits poor performance on multi-turn conversation tasks, like the OpenAI Multi-Round Co-Reference Resolution (MRCR) benchmark [5]. To address this limitation, we perform additional fine-tuning using a diverse dataset. Detailed fine-tuning setup is provided in the Appendix A.2. We also demonstrate the generalization of summarization heads when using different summarization datasets and quantitatively compare the head score distribution with other types of heads in Appendix A.3 and visualize the final KV cache allocation for different types of heads in Appendix A.4

Table 1: Accuracy comparison on MRCR dataset (25% KV cache budget for eviction methods).

	Qwen3-32B				*Qwen2.5-7B-Instruct-1M*				
Method	8k	16k	32k	64k	64k	128k	256k	512k	1M
FullKV	79.22	70.40	68.18	46.42	95.01	96.38	88.6	63.84	43.29
Minference [16]	-	-	-	-	97.11	95.43	82.21	57.39	43.87
AdaKV [4]	26.45	14.7	15.88	12.47	28.34	29.56	21.99	15.28	10.98
DuoAttention [17]	76.19	59.17	60.36	28.37	**85.80**	**89.82**	73.78	39.10	24.73
HeadKV [3]	**78.16**	65.90	**65.72**	34.90	68.52	74.81	66.98	44.36	28.90
Ours	77.05	**68.22**	62.02	**40.13**	85.09	87.19	**83.29**	**48.86**	**34.16**

MRCR The MRCR benchmark [5] tests a model's ability to retrieve information from long, multi-turn dialogues. As shown in Table 1, On Qwen3-32B with Chain-of-Thought (CoT) reasoning, both SGD-KV and HeadKV significantly outperform DuoAttention, suggesting that fine-grained budget allocation is more effective than a binary classification of heads for complex reasoning tasks. As for the fine-tuned Qwen2.5-7B-1M model, SGD-KV consistently outperforms other SOTA head-level allocation methods like HeadKV and AdaKV. While DuoAttention shows strong performance at shorter lengths, our fine-grained allocation strategy proves superior as the context grows, surpassing DuoAttention beyond 128K tokens and demonstrating the most robust performance at the 1M token scale. This highlights the increasing importance of nuanced head prioritization in ultra-long contexts.

ETHIC We also use the ETHIC benchmark [6], which features tasks requiring high information coverage from the input context. Results in Table 2 show that SGD-KV achieves SOTA performance on both models. Notably, with only a 25% KV cache, our method on Qwen3-32B (28.38 Avg.) performs comparably to the Full KV baseline (28.53 Avg.), closing the gap more effectively than any other method. The smaller performance margins between methods on ETHIC, compared to MRCR, are likely attributable to a performance ceiling imposed by the base models' capabilities on these highly complex tasks. Besides, we show the resutls on BABILong [18] benchmark in Appendix A.5.

Table 2: Performance comparison on ETHIC benchmark. "AT", "OG", and "RC" represents the attributing, organizing and recalling sub-tasks, and "Avg." represents the average performance.

	Qwen2.5-7B-Instruct-1M				*Qwen3-32B*			
Method	AT	OG	RC	Avg.	AT	OG	RC	Avg.
Full KV	26.19	17.31	21.45	21.65	31.08	26.11	28.39	28.53
DuoAttention [17]	25.58	**19.55**	13.40	19.51	28.01	20.1	25.35	24.49
HeadKV [3]	26.46	15.48	20.67	20.87	30.90	**25.61**	28.05	28.19
SGD-KV	**27.53**	15.38	**21.12**	**21.34**	**30.94**	25.60	**28.60**	**28.38**

5 Ablation Study: Impact of Query on Token Selection

Table 3: Accuracy comparison on MRCR dataset. Bold values indicate scores that surpass the corresponding Query-Aware baseline results from Table 1.

Mode	Method	8k	16k	32k	64k	128k	256k	512k	1M
Query Unaware	HeadKV	**87.91**	64.4	54.08	48.73	52.62	41.13	26.21	20.38
	Ours	**93.5**	**85.55**	73.02	70.94	77.56	71.06	44.38	33.20
Proxy Query	HeadKV	**90.48**	68.71	60.90	57.52	57.13	50.62	32.41	23.1
	Ours	**97.19**	**89.67**	**82.90**	82.62	83.96	80.65	**50.00**	32.24

Our KV cache compression involves two stages: head-level budget allocation and token-level selection. The token selection step, similar to SnapKV [14], uses the cumulative attention scores generated by a final observation window to identify important tokens. In this study, we ablate the choice of this observation window to understand its impact on performance. We define three conditions: **Query-Aware** (Default): The standard approach used in our main results, where the last 128 tokens (inclduing the real query) guides token selection. **Query-Unaware**: The final question is excluded, and the last 128 tokens of the preceding context are used to guide selection. This simulates a scenario where the specific query is unknown during compression. **Proxy-Query**: A fixed, task-agnostic prompt—*"Segment previous text into unrelated or inconsistent chunk and select keywords for each chunk for later retrieval."* —is used as a universal proxy for the final query.

The results presented in Table 3 illustrates that removing the final question's guidance causes a significant performance degradation for both SGD-KV and HeadKV compared to their Query-Aware performance. However, using the summarization prompt as a Proxy-Query substantially mitigates this performance loss which demonstrates that the summarization task induces a focus on semantically salient information, making our diagnostic prompt an effective and efficient proxy for a real user query. This finding is particularly valuable for applications where the final query is not available beforehand. Finally, we note that across all three conditions, SGD-KV consistently outperforms HeadKV, underscoring the robustness and superior design of our summarization-guided, head-level allocation strategy. Additional ablation studies are provided on different KV cache budgets in Appendix A.6 and on various head configurations in Appendix A.7.

6 Conclusion

In this work, we move beyond conventional, retrieval-based heuristics for identifying specialized attention heads by introducing a new functional class: summarization heads. We proposed a novel chunk-summarization diagnostic task to identify and quantify these heads, which are critical for hierarchical information synthesis. Our resulting framework, SGD-KV, integrates this functional understanding into the KV cache management process, setting a new SOTA on complex, long-context benchmarks while reducing memory usage by up to 75%. Furthermore, our ablation study indicate that a generic summarization prompt can serve as a highly effective proxy query for token selection, mitigating performance degradation when the final question is unavailable. By demonstrating that abstract reasoning roles can be identified and leveraged, our work paves the way for more efficient and interpretable models in the million-token era.

References

[1] An Yang, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoyan Huang, Jiandong Jiang, Jianhong Tu, Jianwei Zhang, Jingren Zhou, Junyang Lin, Kai Dang, Kexin Yang, Le Yu, Mei Li, Minmin Sun, Qin Zhu, Rui Men, Tao He, Weijia Xu, Wenbiao Yin, Wenyuan Yu, Xiafei Qiu, Xingzhang Ren, Xinlong Yang, Yong Li, Zhiying Xu, and Zipeng Zhang. Qwen2.5-1m technical report. *arXiv preprint arXiv:2501.15383*, 2025.

[2] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025. URL https://arxiv.org/abs/2507.06261.

[3] Yu Fu, Zefan Cai, Abedelkadir Asi, Wayne Xiong, Yue Dong, and Wen Xiao. Not all heads matter: A head-level kv cache compression method with integrated retrieval and reasoning. *arXiv preprint arXiv:2410.19258*, 2024.

[4] Yuan Feng, Junlin Lv, Yukun Cao, Xike Xie, and S Kevin Zhou. Ada-kv: Optimizing kv cache eviction by adaptive budget allocation for efficient llm inference. *arXiv preprint arXiv:2407.11550*, 2024.

[5] OpenAI. Mrcr. https://huggingface.co/datasets/openai/mrcr, 2025. Accessed: [2025-05].

[6] Taewhoo Lee, Chanwoong Yoon, Kyochul Jang, Donghyeon Lee, Minju Song, Hyunjae Kim, and Jaewoo Kang. Ethic: Evaluating large language models on long-context tasks with high information coverage. *arXiv preprint arXiv:2410.16848*, 2024.

[7] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

[8] Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Re, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2o: Heavy-hitter oracle for efficient generative inference of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=RkRrPp7GKO.

[9] Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Baobao Chang, Junjie Hu, and Xiao Wen. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling. *arXiv preprint arXiv:2406.02069*, 2024.

[10] Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=EytBpUGB1Z.

[11] Hanlin Tang, Yang Lin, Jing Lin, Qingsen Han, Danning Ke, Shikuan Hong, Yiwu Yao, and Gongyi Wang. Razorattention: Efficient KV cache compression through retrieval heads. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tkiZQlL04w.

[12] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1099. URL https://www.aclweb.org/anthology/P17-1099.

[13] Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. DialogSum: A real-life scenario dialogue summarization dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5062–5074, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.449. URL https://aclanthology.org/2021.findings-acl.449.

[14] 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. *arXiv preprint arXiv:2404.14469*, 2024.

[15] Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

[16] Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H. Abdi, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. MInference 1.0: Accelerating pre-filling for long-context LLMs via dynamic sparse attention. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=fPBACAbqSN.

[17] 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. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=cFu7ze7xUm.

[18] Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 106519–106554. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/c0d62e70dbc659cc9bd44cbcf1cb652f-Paper-Datasets_and_Benchmarks_Track.pdf.

[19] OpenAI. Graphwalks. https://huggingface.co/datasets/openai/graphwalks, 2025. Accessed: [2025-05].

[20] Shibamouli Lahiri. Complexity of Word Collocation Networks: A Preliminary Structural Analysis. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 96–105, Gothenburg, Sweden, April 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/E14-3011.

[21] Akhiad Bercovich, Itay Levy, Izik Golan, et al. Llama-nemotron: Efficient reasoning models, 2025. URL https://arxiv.org/abs/2505.00949.

[22] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

[23] Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.

[24] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3406703. URL https://doi.org/10.1145/3394486.3406703.

[25] Pin-Lun Hsu, Yun Dai, Vignesh Kothapalli, Qingquan Song, Shao Tang, Siyu Zhu, Steven Shimizu, Shivam Sahni, Haowen Ning, Yanning Chen, and Zhipeng Wang. Liger-kernel: Efficient triton kernels for LLM training. In *Championing Open-source DEvelopment in ML Workshop @ ICML25*, 2025. URL https://openreview.net/forum?id=36SjAIT42G.

[26] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

[27] Chenxin An, Fei Huang, Jun Zhang, Shansan Gong, Xipeng Qiu, Chang Zhou, and Lingpeng Kong. Training-free long-context scaling of large language models, 2024.

[28] Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023. URL https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm.

[29] Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5409. URL https://www.aclweb.org/anthology/D19-5409.

[30] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *ArXiv*, abs/1808.08745, 2018.

[31] Faisal Ladhak, Esin Durmus, Claire Cardie, and Kathleen McKeown. WikiLingua: A new benchmark dataset for cross-lingual abstractive summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4034–4048, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.360. URL https://aclanthology.org/2020.findings-emnlp.360.

A Appendix

A.1 KV Cache Budget Allocation & Redistribution

Duo to the task complexity requirements and the goal of minimal performance degradation, we employ a relatively large KV cache budget, which can result in $S_h \cdot L \cdot H \cdot R > 1$ for extremely high-importance heads. We address this through a water-filling inspired redistribution algorithm:

Algorithm 1 Summarization-Aware Budget Redistribution

Input: Normalized head importance scores S_h for all heads h, M is the middle sequence length
Output: Redistributed budget allocation $\tilde{b}_h$ for all heads

1: Initialize b_h for all heads using Equation (6)
2: Compute excess budget: $E = \sum h : b_h > M(b_h - M)$
3: Cap over-allocated heads: $\tilde{b}_h \leftarrow \min(b_h, M)$ for all h
4: Sort remaining heads by S_h in descending order
5: Distribute E to top-scoring heads with available capacity

This approach prioritizes high-importance summarization heads while maintaining budget constraints, ensuring that critical attention mechanisms receive adequate KV cache allocation for effective long-context summarization.

A.2 Fine-tuning details

Fine-tuning Dataset. The fine-tuning dataset comprises: 10K synthetic MRCR samples, 20K synthetic OpenAI GraphWalks samples [19], and 25K BABILong fine-tuning samples [18]. To prevent overfitting, we incorporate 10K samples from the Gutenberg dataset [20] and 10K samples from the Llama-Nemotron-Post-Training-Dataset-v1.1 [21] for regularization.

Fine-tuning Recipe. We use Llama Factory library [22] as the framework to supervised fine-tune the Qwen2.5-7B-Instruct-1M [1] model. We use full-parameter fine-tuning, cosine learnting schedule with inital learning rate as 1.0×10^{-5}, warmming-up ratios is 0.1, total batch size is 128 and total training epochs is 2. To reduce GPU memory we use flash attention [23], DeepSpeed with stage 0 [24] and Liger kernel [25].

Baselines. We compare SGD-KV against four strong baselines representing different approaches to KV cache optimization: (i) DuoAttention [17] finetunes a gate function to binary classify heads into retrieval heads (receiving full KV cache) and streaming heads (retaining only recent tokens and attention sink tokens). Since we do not finetune LLMs on retrieval datasets, we use R2 scores to binary partition heads based on available KV cache budget. (ii) AdaKV [4] pioneered head-level KV cache budget allocation by analyzing top-K attention values across heads during inference. (iii) HeadKV [3] introduced retrieval-reasoning heads, computing offline importance scores for each head and allocating KV cache budget proportionally based on these scores. (iv) MInference [16] determines optimal attention patterns for each head offline and dynamically constructs sparse indices based on assigned patterns during inference. For fair comparison with DuoAttention, we adapt their binary classification approach by using R2 scores to partition heads into retrieval and streaming categories according to our KV cache budget constraints, rather than performing finetuning. For the MInference, since we directly use the VLLM [26] to get the results, it always integrates with the Dual Chunk Attention (DCA) [27]

Configuration. For all baseline methods, we retain the first 1024 tokens as the attention sink, while maintaining a context window of 1024 recent tokens for the Qwen2.5-7B-1M model. For the larger Qwen3-32B model, both the sink size and window size are reduced to 128 tokens. All KV cache allocations occur before repeating the KV values, meaning that the KV cache memory savings are built upon the GQA. We also found that if we rerun the identification process for the models after SFT, the HeadKV method with new configuration got worse worse performance. Therefore, we used the configuration of the original checkpoint for HeadKV, DuoAttetnion and our method.

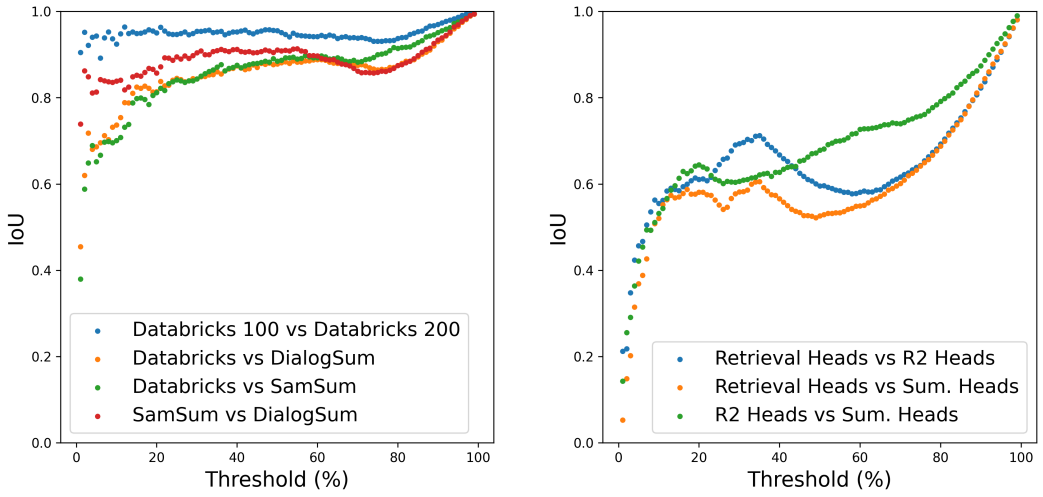

Figure 2: The comparisons of IoU scores between summarization heads with different datasets (left) and different types of heads (right).

A.3 Quantitatively Analysis of the Stability and Generalization of the Summarization Heads

At first, we investigate whether summarization heads exhibit consistent patterns across different datasets and tasks. Since we do not impose hard thresholds to classify heads as "summarization heads," we maintain continuous importance scores for all heads, following the approach of HeadKV [3].

To quantitatively assess distributional consistency, we rank all attention heads by their importance scores in descending order and compute Intersection over Union (IoU) between ranked lists from different experimental conditions:

$$\text{IoU@}k = \frac{|\mathcal{H}_1^{(k)} \cap \mathcal{H}_2^{(k)}|}{|\mathcal{H}_1^{(k)} \cup \mathcal{H}_2^{(k)}|} \tag{3}$$

where $\mathcal{H}_i^{(k)}$ represents the top-k heads from ranking i.

Fig. 2 demonstrates high consistency in summarization head identification. The blue curve shows IoU between two disjoint 100-sample subsets from Databricks Dolly dataset [28], achieving near-perfect overlap (IoU > 0.9), indicating robust head identification within datasets. Cross-dataset analysis reveals substantial generalization: Dolly vs. DialogSum [13] (orange), Dolly vs. SAMSum (green), and DialogSum vs. SAMSum (red) maintain high IoU scores, particularly for top-ranked heads. Notably, dialogue-based datasets (DialogSum and SAMSum) exhibit stronger similarity, suggesting task-specific head specialization.

For the final configuration of the summarization heads, we use the average scores from the 6 datasets, DialogSum dataset [13], SAMSum dataset [29], CNN/Dailymail dataset [12], Extreme Summarization (XSum) dataset [30], and the summarization part of the Databricks Dolly dataset [28] and WikiLingua [31] dataset.

We also compare summarization heads with retrieval heads and recently proposed R2 heads. The right panel of Fig. 2 shows that while the top 20% of heads overlap significantly across head types, the 20-60% percentile range reveals distinct preferences, confirming that summarization heads capture unique attention patterns beyond simple retrieval mechanisms.

A.4 Visualization of Different Types of Heads

As shown in Fig. 3, we visualize the KV cache budget allocation across different attention heads. Since we evict KV cache entries before they repeat, the visualization displays 4 rows representing head indices and 28 columns representing layer indices. Both R2 heads and summarization heads

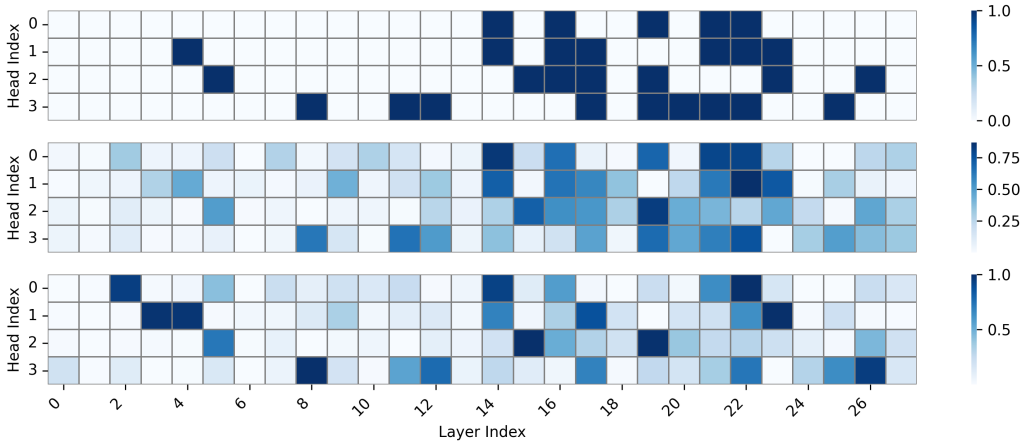

Figure 3: The visualization of the confusion matrix of the head scores of DuoAttention (top), HeadKV (center) and ours (bottom) in Qwen2.5-7B-Instruct-1M model. Y axis is the index of the heads, 4 in total, and the X axis is the index of the layers, 28 in total.

exhibit similar patterns in their KV cache distributions, indicating that retrieval capability serves as a foundational ability underlying our summarization heads, which encompass both simple retrieval mechanisms and complex information aggregation processes. There are also few distinct pattern between R2 heads and summarization heads, our method allocates relatively more budget to earlier layers, which are essential for complex reasoning and summarization tasks. This allocation strategy proves crucial because when faced with complex queries, models may be unable to identify all necessary tokens before generation. In such scenarios, methods like HeadKV may inadvertently evict key information that lacks direct surface-level connections to the query but remains semantically relevant for comprehensive understanding.

A.5 BABILong Results

We also evaluate the methods on the BABILong benchmark [18], which is specifically designed to test the retrieval and reasoning capabilities of LLMs. As shown in Table 4, despite HeadKV's use of similar question formats to identify R2 heads, it shows negligible differences compared to our summarization heads and achieves similar accuracy to the full KV cache model. This further validates the generalization ability of our summarization heads. Furthermore, both HeadKV and our method outperform DuoAttention, suggesting that when the KV cache budget is sufficient for head-level KV eviction methods, they typically deliver superior performance compared to DuoAttention methods under the same KV cache constraints.

Table 4: Accuracy comparison between different methods on BABILong dataset (average over QA1 to QA5) from 32K to 1M sequence length using fine-tuned Qwen2.5-7B-Instruct-1M.

Method	KV Budget (%)	32k	64k	128k	256k	512k	1M
FullKV	100	95.4	96.6	97.2	95.6	96.2	94.6
DAC+Minference	-	95.8	96.2	97.6	96.0	87.2	76.6
DuoAttention (static)	25	89.2	92.8	92.0	95.6	88.4	88.2
HeadKV	25	94.8	96.0	96.6	95.8	96.0	94.2
SGD-KV (Ours)	25	94.4	95.8	96.6	95.6	95.8	94.2

A.6 Performance under Different KV Cache Budget

The results in Fig.4 demonstrate the robust performance of SGD-KV, which consistently surpasses both AdaKV and HeadKV across almost all KV cache budgets. The performance margin narrows only in extreme cases (KV cache budgets <15% or >50%). Moreover, when the cache budget exceeds 50%, both SGD-KV and HeadKV begin to outperform MInference, particularly for context lengths

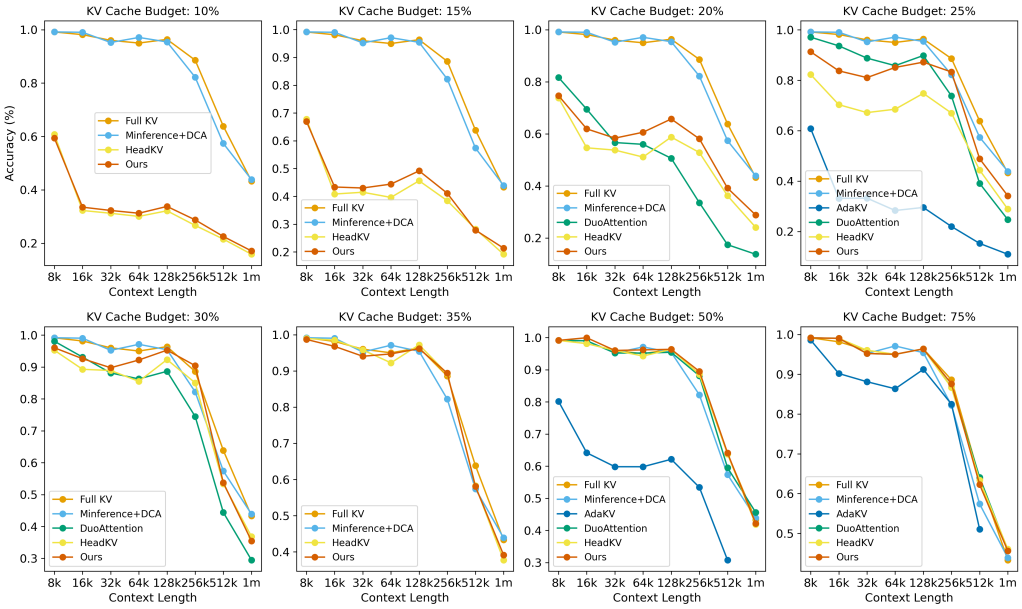

Figure 4: The comparisons of accuracy on the MRCR dataset under Different KV Cache Budget.

longer than 256K. This finding aligns with our results on the BABILong benchmark, confirming that given an adequate cache, head-level eviction strategies are more effective than token-level approaches like DuoAttention under identical budget constraints. Due to time and computational constraints, we omitted some results for DuoAttention (at 10%, 15%, 35% budgets) and AdaKV (at 10%, 15%, 20%, 30%, 35% budgets). Nevertheless, the trends established by the available data strongly suggest that these omissions do not alter our overall conclusions.

A.7 Head Configurations Variants

In this section, we conduct a series of experiments with different configurations of the summarization heads and the R2 heads to evaluate the effectiveness of both types. Due to time and computational resource constraints, we omit some results on the 1M context length. However, since the observed trends are generally consistent across context lengths, the absence of these results does not affect our overall conclusions. We introduce an additional column in Table 5, Full KV Heads (%), which denotes the proportion of attention heads granted access to the full KV cache.

In the first (top) part of Table 5, we report results for the summarization-head variant of DuoAttention (denoted DuoAttention (Sum.)). Within the DuoAttention framework, using summarization heads consistently underperforms compared to R2 heads. This suggests that R2 heads prioritize ranking heads by retrieval capability, whereas summarization heads primarily focus on fine-grained allocation of KV cache across heads. We further evaluate DuoAttention (Sum., Reverse), where the bottom 75% of heads (with low summarization scores) are assigned full KV cache access, while the top 25% only retain access to initial and recent tokens. The substantial performance drop demonstrates that constraining the KV cache of high-summarization-score heads is far more detrimental than constraining low-score heads, thereby validating the informativeness of the summarization score.

In the second (middle) part of Table 5, we present results for SGD-KV (Reverse), where the KV cache allocation is inverted. For instance, a head originally assigned 10% of the KV cache is instead allocated 90%. The third row shows that even with $3\times$ KV cache, SGD-KV (Reverse) performs significantly worse than SGD-KV, further underscoring the importance of score-guided distribution. We also report results for SGD-KV + HeadKV, obtained by averaging the normalized R2 scores and summarization scores. As expected, its performance lies between HeadKV and SGD-KV.

In the last row of the second part, we evaluate SGD-KV (thr), where heads with summarization scores below the mean are set to zero before redistribution. This strategy biases cache allocation toward

higher-scoring heads, yielding a larger proportion of full KV heads. Results show improved accuracy for context lengths up to 64k, but degraded performance beyond 128k. This indicates that heads with relatively low summarization scores remain important for ultra-long context modeling.

In the last (bottom) part of Table 5, we investigate the effect of leveraging summarization scores to guide attention score aggregation in models with GQA. Take the Qwen2.5-7B-Instruct-1M model as an example: it has 28 attention heads but only 4 key-value (KV) heads, corresponding to a group size of 7. When computing summarization scores, we obtain a score for each individual attention head, but these must be aggregated into a single score per KV head. The most straightforward approaches are averaging or taking the maximum. In our experiments, we adopt the maximum, as it consistently outperforms the mean operation by a small margin.

Based on this, we evaluate three new configurations:

- **Ipt., Max**: multiply the attention scores by the summarization scores for each head, followed by max-pooling within each group.
- **Ipt., Mean**: multiply the attention scores by the summarization scores for each head, followed by mean-pooling within each group.
- **Ipt., Only**: use only the summarization scores. For instance, if head 3 has the highest summarization score within a group of 7, then all 7 heads inherit the attention scores of head 3.

As shown in the results, all three methods improve performance to varying degrees. Among them, Ipt., Max achieves the best overall accuracy from 8k to 512k context lengths, indicating that incorporating summarization scores into attention score aggregation is beneficial for GQA models.

Table 5: Accuracy comparison between different methods on OpenAI MRCR dataset from 8K to 512K sequence length using fine-tuned Qwen2.5-7B-Instruct-1M.

Method	Full KV Heads(%)	8k	16k	32k	64k	128k	256k	512k
FullKV	100	99.16	98.16	96.02	95.01	96.38	88.6	63.84
DuoAttention (R2)	25	**97.09**	**93.65**	88.78	85.80	89.82	73.78	39.10
DuoAttention (Sum.)	25	92.11	82.25	72.86	70.56	79.55	65.24	28.45
DuoAttention (Sum., Reverse)	75	45.62	14.13	12.75	8.5	5.69	5.96	6.26
HeadKV	0.0	82.27	70.31	67.18	68.52	74.81	66.98	44.36
SGD-KV	3.57	91.30	83.73	81.06	85.09	87.19	83.29	48.86
SGD-KV (Reverse)	0.0	47.21	17.31	15.85	11.31	8.21	8.62	8.61
SGD-KV + HeadKV	1.79	88.12	77.55	75.34	75.52	82.53	77.32	45.28
SGD-KV (thr)	16.96	95.51	89.62	82.49	85.43	86.24	70.03	32.22
SGD-KV (Ipt., max)	3.57	95.5	91.49	**89.00**	**87.74**	**91.16**	**87.75**	**57.23**
SGD-KV (Ipt., mean)	3.57	94.37	89.55	87.34	85.37	90.55	85.37	55.57
SGD-KV (Ipt., only)	3.57	95.71	86.35	86.38	87.30	90.63	82.09	49.91