Q-FILTERS: LEVERAGING QUERY-KEY GEOMETRY FOR EFFICIENT KEY-VALUE CACHE COMPRESSION

Nathan Godey[1,2*] **Alessio Devoto**[3] **Yu Zhao**[4] **Simone Scardapane**[3]

Pasquale Minervini[4] **Éric de la Clergerie**[1] **Benoît Sagot**[1]

[1]Inria, Paris [2]Sorbonne Université [3]Sapienza University of Rome [4]University of Edinburgh

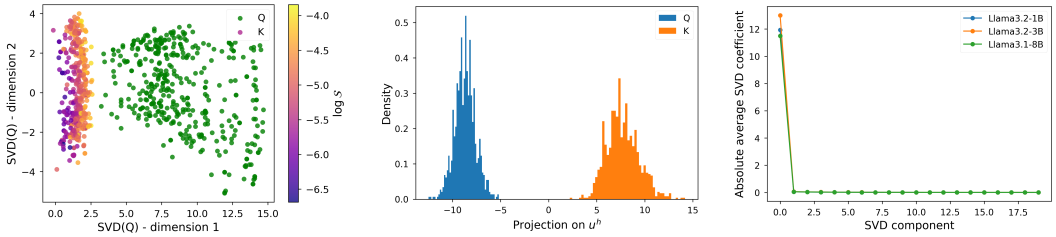
ABSTRACT

Autoregressive language models rely on a Key-Value (KV) Cache, which avoids re-computing past hidden states during generation, making it faster. As model sizes and context lengths grow, the KV Cache becomes a significant memory bottleneck, which calls for compression methods that limit its size during generation. In this paper, we discover surprising properties of Query (Q) and Key (K) vectors that allow us to efficiently approximate attention scores without computing the attention maps. We propose Q-Filters, a training-free KV Cache compression method that filters out less crucial Key-Value pairs based on a single context-agnostic projection. Contrarily to many alternatives, Q-Filters is compatible with FlashAttention, as it does not require direct access to attention weights. Experimental results in long-context settings demonstrate that Q-Filters is competitive with attention-based compression methods such as SnapKV in retrieval tasks while consistently outperforming efficient compression schemes such as Streaming-LLM in generation setups. Notably, Q-Filters achieves a 99% accuracy in the needle-in-a-haystack task with a $\times 32$ compression level while reducing the generation perplexity drop by up to 65% in text generation compared to Streaming-LLM.

1 INTRODUCTION

The performance of Large Language Models (LLMs) as autoregressive text-generation systems relies on the effectiveness of the Transformer architecture (Vaswani et al., 2017). Recently, long-context models such as Gemini-Pro-1.5 (Reid et al., 2024), Claude-3 (Anthropic, 2024), GPT-4 (Achiam et al., 2023), and Llama-3.1 (Dubey et al., 2024) have demonstrated the ability to process hundreds of thousands of tokens. However, processing such long sequences comes with significant challenges, as it may lead to higher decoding latency and memory saturation. As the context length grows, each inference step involves storing an increasingly large context from GPU memory in the form of the KV Cache, creating a memory bottleneck that hinders efficient inference (Fu, 2024). To address this issue, KV Cache compression methods aim to reduce the size of this past-context representations storage by removing or merging Key-Value pairs, thereby alleviating memory bottlenecks. While KV Cache compression techniques have gained popularity, many approaches require fine-tuning or re-training the underlying models (Nawrot et al., 2024; Ainslie et al., 2023; DeepSeek-AI et al., 2024), which limits their applicability in real-world deployment scenarios. Training-free methods have also been proposed, but they often rely on access to attention weights to evaluate the importance of Key-Value pairs (Xiao et al., 2024; Li et al., 2024), making them incompatible with the widely adopted efficient attention algorithm FlashAttention (Dao, 2024). These methods usually require a partial re-computation of the attention matrices, which leads to a time and memory overhead. Hence, these algorithms are often used to compress prompts before generating answers and are not ideally suited for memory-constrained generation.

*Contact: nathan.godey@inria.fr



(a) Layer 0, Head 21 (b) Layer 14, Head 5 ($\epsilon = -1$) (c) SVD absolute avg. coefficients

Figure 1: Left and center: distributions of the projections of Q^h and K^h on u^h for Llama-3.1-8B. Right: estimates of $\left|\mathbb{E}_i(\langle Q_i^h, v_m\rangle)\right|$ where v_m are the right vectors from the SVD of a set of Q^h representations from different Llama models, averaged over all layers and heads.

In this work, we propose Q-Filters, a training-free KV Cache compression method that uses the geometric properties of **Q**ueries and Keys to **filter** out the less important Key-Value pairs. Our approach achieves competitive results across synthetic tasks and pure generation cases while maintaining compatibility with FlashAttention and, thus, better time efficiency.

Analysing the properties of queries (Q) and Keys (K) distributions, we find that *a single direction, spanned by the principal component of Q, encodes an input selection process for each head*. Identifying this direction allows us to efficiently estimate which inputs are mostly ignored by a given head and can thus be discarded with minimal performance loss. Interestingly, we find that this direction is context-agnostic, *i.e.,* the directions we identify in different contexts are highly consistent. Leveraging this property, we calculate lightweight projections, which we refer to as Q-Filters, based on a small held-out calibration dataset only once for every model, incurring minimal computational overhead. At inference time, we use Q-Filters to project Keys in the pre-computed direction to estimate the importance of Key-Value pairs without accessing attention scores, and we prune the KV Cache accordingly. This makes our method faster than most KV Cache compression alternatives that use attention scores to estimate the importance of the KV pairs.

Additionally, our method is training-free, requiring only a very short initial calibration, and we show it can be easily applied to a variety of decoder-only language models. We validate our method on a wide set of tasks, ranging from language modelling to in-context learning and long-context tasks, achieving competitive performance even with 32x compression ratios.

2 Method

2.1 Exploring the Query-Key Geometry

In Devoto et al. (2024), the authors examined a relationship between basic characteristics of the Key representations and attention score distributions. Notably, they observe a negative correlation between the average attention weight given to a position and the L_2-norm of the K_t^h vector at that position (where h is the head index, and t the position in the sequence). Leveraging this observation, they propose to compress the KV Cache by selecting the KV pairs for which $||K_t^h||_2$ is the smallest. Using this simple heuristic, they are able to reach $\times 2$ compression ratios without altering the retrieval and modelling performance of the models they study.

Godey et al. (2024) show that the distributions of Q_t^h and K_t^h are *anisotropic*, i.e. they do not uniformly occupy $\mathbb{R}^{d_H}$. They observe that both distributions "drift away" from the origin as training progresses. Crucially, this drift occurs along parallel directions in $\mathbb{R}^{d_H}$, so that the dot product between mean Q_t^h and mean K_t^h representations tends to increase in absolute value and to be either positive or negative for different heads.

Motivated by Devoto et al. (2024) and Godey et al. (2024), we propose to further explore some geometrical properties of Q^h and K^h vectors and their implications for unnormalized attention logits $Q^h(K^h)^T$. First, we expose the *joint* anisotropy of Q^h and K^h in Figure 1b. Empirically, we observe a head-dependent direction u^h so that projections of Q_t^h and K_t^h in this direction are each of consistent sign. This direction can be identified by performing a Singular Value Decomposition (SVD)

on a set of Q^h representations and setting $u^h = \pm v_1$ where v_1 is the right-vector corresponding to the highest singular value, and where the sign is chosen to make most Q_t^h (if not all) project to a positive value.

Interestingly, as shown in Figure 1c, v_1 is the only component of the SVD that projects Q^h representations to a non-null average coefficient. The intuitive consequence of these two observations is that if a given K_t^h has a strong (negative) projection along u^h, then the $\langle Q_i^h, K_t^h \rangle$ dot products should *all* be negatively affected, which results in less important attention weights towards K_t^h. Since v_1 is the only direction that encodes such a phenomenon, one can derive an approximation of the average attention score received by K_t^h based on its projection on $v_1 = u^h$, which is summarized in Proposition 2.1.

Proposition 2.1 (proof in Appendix B)**.** *Under Hypothesis A.1 and Hypothesis A.2, we have:*

$$\mathbb{E}_{Q_i^h}(\langle Q_i^h, K_j^h \rangle) \approx \kappa^h \langle K_j^h, u^h \rangle$$

where κ^h is a positive constant.

Intuitively, projecting K_t^h along the anisotropic direction u^h gives us an estimate of the attention logits that involve K_t^h up to a positive multiplicative constant κ^h.

The resulting v_1 vectors, that we refer to as Q-Filters, allow to estimate, up to a sign, which Key-Value pairs are worth storing for each head along generation.

2.2 Q-Filters

Based on Proposition 2.1, we can design a KV Cache compression scheme that consists of the following steps:

1. For a given model, retrieve its Q-Filters, which can be obtained with the following procedure:
   (a) Gather Q^h representations by passing samples through the model;
   (b) Compute the SVD of the gathered representations at each layer and head;
   (c) Obtain the *positive* right vector (or Q-Filter) for each head $v_1^+ = \text{sgn}(\mathbf{1} u_1^T) v_1$.
2. At inference, for each head, discard the K_t^h with the lowest $\langle K_t^h, v_1^+ \rangle$ value.

In the case of Grouped-Query Attention or GQA (Ainslie et al., 2023), we simply average the Q-Filters for each group of Query representations.

We bring the attention of the reader to the fact that this method only requires a single preparation step following training for a given model. The Q-Filters are entirely context-agnostic and rely on inherent properties of the Query and Key latent spaces. In the rest of this article, we use a subset of the Pile dataset (Gao et al., 2020) to compute the Q-Filters and discuss the choice of the dataset and of the number of necessary SVD samples in Appendix H.

In Figure 11, we observe that the Q-Filters heuristic is noticeably more correlated with the attention score for most heads compared to the L_2-norm metric. As such, ordering the Key-Value pairs using the Q-Filters heuristic should allow us to select more relevant pairs than using the method from Devoto et al. (2024) - that we will call K-norm for the sake of simplicity.

3 Experiments

We validate our method both on memory-constrained language modelling and on long-context retrieval tasks (e.g. needle-in-a-haystack). Additionally, we test our method on the Ruler dataset Hsieh et al. (2024), which is specifically designed to test the model's long context modelling abilities. We test Q-Filters on Llama-3.1-8B, Llama-3.1-70B Dubey et al. (2024), and Qwen-2.5-7B Qwen et al. (2025), but the method can be easily used for any pre-trained decoder-only LLM. We compare Q-Filters with several KV Cache compression methods. These include StreamingLLM (Xiao et al., 2024), which focuses on language modelling by always retaining the initial tokens of the sequence. We also compare with SnapKV (Li et al., 2024), which performs compression based on attention scores from the final portion of the prompt, making it particularly suitable for compression of large

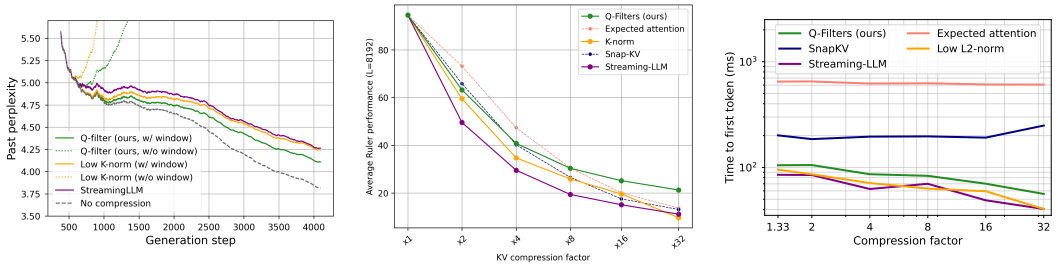


Figure 2: Left (a): Llama-3.1-8B Language Modelling performance in the streaming compression setup. Center (b): Average performance on Ruler (8192) for Llama-3.1-8B-Instruct. Right (c): First token latency across KV Cache compression methods of Llama-3.2-8B with a length of 64k prompt.

prompts. Additionally, we compare against preserving low-L_2 norm tokens (Devoto et al., 2024) and the recent ExpectedAttention (Jegou & Jeblick, 2024).

We show results for different tasks in Figure 2. In Language modelling, we observe that Q-Filters consistently achieves the lowest perplexity among compression schemes, even for very long contexts. In the Ruler dataset Hsieh et al. (2024), we test the model's score for several compression factors ranging from $2\times$ to $32\times$. While for some lower compression factors, we find performance on par with other methods, Q-Filters achieve the highest score with the strongest compression factor of $32\times$, demonstrating the method's effectiveness at high compression rates. We provide detailed results in table Table 1 and further discussion on additional benchmarks in Appendix G.

Compression method	FA-compatible	CWE	FWE	Multi-Key	Multi-Query	Multi-Value	Single	QA	VT	Average
SnapKV	✗	**88.7**	89.0	15.1	29.6	28.8	68.7	42.8	83.2	50.5
Expected Attention	✗	70.0	79.3	12.0	**59.7**	37.8	31.2	44.2	96.3	43.2
Streaming-LLM	✓	53.8	**93.4**	14.1	16.8	16.7	15.7	**62.3**	15.8	31.6
K-Norm	✓	22.9	74.8	8.7	16.6	25.8	55.9	20.6	32.0	31.3
Q-Filters (ours)	✓	82.5	80.2	**22.9**	49.1	**60.6**	**71.1**	37.6	**100**	**56.1**

Table 1: Results on the Ruler-4096 dataset for Llama-3.1-70B-Instruct with an $8\times$ compression ratio. The second column indicates compatibility with FlashAttention.

3.1 Throughput and Scalability

Our approach is more efficient than many KV Cache compression methods, as it estimates the relevance of a K^h representation *without materializing the attention maps*. This property makes it compatible with memory-efficient self-attention implementations such as FlashAttention (Dao, 2024). During inference, Q-Filters maintains the same theoretical time complexity as the K-norm method (Devoto et al., 2024), since computing a norm and a scalar product require a comparable number of floating-point operations.

By avoiding the explicit computation of attention scores, our method achieves lower inference latency compared to existing approaches. To quantify this efficiency, we measure the *Time to First Token* across different methods in Section 3. Time to First Token (TTFT) refers to the latency between submitting a prompt and receiving the first generated token. This metric is particularly relevant in scenarios where fast response times are critical, such as interactive AI applications. Compressing the KV Cache directly impacts TTFT: by reducing the memory footprint of the KV Cache, it allows a larger portion of the prompt context to fit within fast-access memory, minimizing memory swapping overhead. As a result, compression techniques that efficiently manage the KV Cache should significantly reduce initial response latency. Notably, our experiments show that Q-Filters maintain this performance advantage even as the sequence length increases, suggesting better scalability compared to methods that require explicit attention computation.

4 Conclusion

We introduced Q-Filters, a novel training-free method for KV Cache compression. We show that projecting the Key representations on the main SVD component of the Query vectors results in an accurate approximation of the attention scores. Q-Filters is extremely efficient and is compatible with FlashAttention as it does not require accessing the attention scores. We validated our method on several tasks (Language modelling, NIAH, Ruler) and models up to 70B parameters, and showed competitive performance with respect to more costly state-of-the-art KV Cache compression methods.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Ainslie, J., Lee-Thorp, J., de Jong, M., Zemlyanskiy, Y., Lebron, F., and Sanghai, S. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=hmOwOZWzYE.

Anthropic, A. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 2024.

Cai, Z., Zhang, Y., Gao, B., Liu, Y., Liu, T., Lu, K., Xiong, W., Dong, Y., Chang, B., Hu, J., and Xiao, W. Pyramidkv: Dynamic kv cache compression based on pyramidal information funneling, 2024. URL https://arxiv.org/abs/2406.02069.

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

DeepSeek-AI, Liu, A., Feng, B., Wang, B., Wang, B., Liu, B., Zhao, C., Dengr, C., Ruan, C., Dai, D., Guo, D., Yang, D., Chen, D., Ji, D., Li, E., Lin, F., Luo, F., Hao, G., Chen, G., Li, G., Zhang, H., Xu, H., Yang, H., Zhang, H., Ding, H., Xin, H., Gao, H., Li, H., Qu, H., Cai, J. L., Liang, J., Guo, J., Ni, J., Li, J., Chen, J., Yuan, J., Qiu, J., Song, J., Dong, K., Gao, K., Guan, K., Wang, L., Zhang, L., Xu, L., Xia, L., Zhao, L., Zhang, L., Li, M., Wang, M., Zhang, M., Zhang, M., Tang, M., Li, M., Tian, N., Huang, P., Wang, P., Zhang, P., Zhu, Q., Chen, Q., Du, Q., Chen, R. J., Jin, R. L., Ge, R., Pan, R., Xu, R., Chen, R., Li, S. S., Lu, S., Zhou, S., Chen, S., Wu, S., Ye, S., Ma, S., Wang, S., Zhou, S., Yu, S., Zhou, S., Zheng, S., Wang, T., Pei, T., Yuan, T., Sun, T., Xiao, W. L., Zeng, W., An, W., Liu, W., Liang, W., Gao, W., Zhang, W., Li, X. Q., Jin, X., Wang, X., Bi, X., Liu, X., Wang, X., Shen, X., Chen, X., Chen, X., Nie, X., Sun, X., Wang, X., Liu, X., Xie, X., Yu, X., Song, X., Zhou, X., Yang, X., Lu, X., Su, X., Wu, Y., Li, Y. K., Wei, Y. X., Zhu, Y. X., Xu, Y., Huang, Y., Li, Y., Zhao, Y., Sun, Y., Li, Y., Wang, Y., Zheng, Y., Zhang, Y., Xiong, Y., Zhao, Y., He, Y., Tang, Y., Piao, Y., Dong, Y., Tan, Y., Liu, Y., Wang, Y., Guo, Y., Zhu, Y., Wang, Y., Zou, Y., Zha, Y., Ma, Y., Yan, Y., You, Y., Liu, Y., Ren, Z. Z., Ren, Z., Sha, Z., Fu, Z., Huang, Z., Zhang, Z., Xie, Z., Hao, Z., Shao, Z., Wen, Z., Xu, Z., Zhang, Z., Li, Z., Wang, Z., Gu, Z., Li, Z., and Xie, Z. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024. URL https://arxiv.org/abs/2405.04434.

Dehghani, M., Djolonga, J., Mustafa, B., Padlewski, P., Heek, J., Gilmer, J., Steiner, A. P., Caron, M., Geirhos, R., Alabdulmohsin, I., Jenatton, R., Beyer, L., Tschannen, M., Arnab, A., Wang, X., Riquelme Ruiz, C., Minderer, M., Puigcerver, J., Evci, U., Kumar, M., Steenkiste, S. V., Elsayed, G. F., Mahendran, A., Yu, F., Oliver, A., Huot, F., Bastings, J., Collier, M., Gritsenko, A. A., Birodkar, V., Vasconcelos, C. N., Tay, Y., Mensink, T., Kolesnikov, A., Pavetic, F., Tran, D., Kipf, T., Lucic, M., Zhai, X., Keysers, D., Harmsen, J. J., and Houlsby, N. Scaling vision transformers to 22 billion parameters. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 7480–7512. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/dehghani23a.html.

Devoto, A., Zhao, Y., Scardapane, S., and Minervini, P. A simple and effective l_2 norm-based strategy for KV cache compression. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp.

18476–18499, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1027. URL https://aclanthology.org/2024.emnlp-main.1027.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., Rao, A., Zhang, A., Rodriguez, A., Gregerson, A., Spataru, A., Rozière, B., Biron, B., Tang, B., Chern, B., Caucheteux, C., Nayak, C., Bi, C., Marra, C., McConnell, C., Keller, C., Touret, C., Wu, C., Wong, C., Ferrer, C. C., Nikolaidis, C., Allonsius, D., Song, D., Pintz, D., Livshits, D., Esiobu, D., Choudhary, D., Mahajan, D., Garcia-Olano, D., Perino, D., Hupkes, D., Lakomkin, E., AlBadawy, E., Lobanova, E., Dinan, E., Smith, E. M., Radenovic, F., Zhang, F., Synnaeve, G., Lee, G., Anderson, G. L., Nail, G., Mialon, G., Pang, G., Cucurell, G., Nguyen, H., Korevaar, H., Xu, H., Touvron, H., Zarov, I., Ibarra, I. A., Kloumann, I. M., Misra, I., Evtimov, I., Copet, J., Lee, J., Geffert, J., Vranes, J., Park, J., Mahadeokar, J., Shah, J., van der Linde, J., Billock, J., Hong, J., Lee, J., Fu, J., Chi, J., Huang, J., Liu, J., Wang, J., Yu, J., Bitton, J., Spisak, J., Park, J., Rocca, J., Johnstun, J., Saxe, J., Jia, J., Alwala, K. V., Upasani, K., Plawiak, K., Li, K., Heafield, K., Stone, K., and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. doi: 10.48550/ARXIV.2407.21783. URL https://doi.org/10.48550/arXiv.2407.21783.

Fu, Y. Challenges in deploying long-context transformers: A theoretical peak performance analysis. *arXiv preprint arXiv:2405.08944*, 2024.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., and Leahy, C. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Godey, N., Clergerie, É., and Sagot, B. Anisotropy is inherent to self-attention in transformers. In Graham, Y. and Purver, M. (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 35–48, St. Julian's, Malta, March 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.eacl-long.3.

Guo, Z., Kamigaito, H., and Watanabe, T. Attention score is not all you need for token importance indicator in KV cache reduction: Value also matters. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 21158–21166, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1178. URL https://aclanthology.org/2024.emnlp-main.1178/.

Hsieh, C.-P., Sun, S., Kriman, S., Acharya, S., Rekesh, D., Jia, F., Zhang, Y., and Ginsburg, B. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Jegou, S. and Jeblick, M. Kvpress, 2024. URL https://github.com/kvpress.

Li, Y., Huang, Y., Yang, B., Venkitesh, B., Locatelli, A., Ye, H., Cai, T., Lewis, P., and Chen, D. SnapKV: LLM knows what you are looking for before generation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=poE54GOq2l.

Nawrot, P., Łańcucki, A., Chochowski, M., Tarjan, D., and Ponti, E. Dynamic memory compression: Retrofitting LLMs for accelerated inference. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=tDRYrAkOB7.

OLMo, T., Walsh, P., Soldaini, L., Groeneveld, D., Lo, K., Arora, S., Bhagia, A., Gu, Y., Huang, S., Jordan, M., Lambert, N., Schwenk, D., Tafjord, O., Anderson, T., Atkinson, D., Brahman, F., Clark, C., Dasigi, P., Dziri, N., Guerquin, M., Ivison, H., Koh, P. W., Liu, J., Malik, S., Merrill, W., Miranda, L. J. V., Morrison, J., Murray, T., Nam, C., Pyatkin, V., Rangapur, A., Schmitz, M., Skjonsberg, S., Wadden, D., Wilhelm, C., Wilson, M., Zettlemoyer, L., Farhadi, A., Smith, N. A., and Hajishirzi, H. 2 olmo 2 furious, 2025. URL https://arxiv.org/abs/2501.00656.

Qwen, :, Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Tang, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., and Qiu, Z. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Reid, M., Savinov, N., Teplyashin, D., Lepikhin, D., Lillicrap, T., Alayrac, J.-b., Soricut, R., Lazaridou, A., Firat, O., Schrittwieser, J., et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

Shazeer, N. Fast transformer decoding: One write-head is all you need. 2019. URL https://arxiv.org/abs/1911.02150.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

Zhang, Z., Sheng, Y., Zhou, T., Chen, T., Zheng, L., Cai, R., Song, Z., Tian, Y., Ré, C., Barrett, C., et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

A OBSERVATIONS

Hypothesis A.1 (Joint anisotropy)**.** There exist $u^h \in \mathbb{S}^{d_H-1}$ and $\epsilon = \pm 1$ such that

$$\mathbb{E}\left(\langle Q_i^h, u^h \rangle\right) > 0 \quad \text{and} \quad \mathbb{E}\left(\langle K_j^h, \epsilon u^h \rangle\right) > 0,$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product.

To validate Hypothesis A.1, we compute the Singular Value Decomposition (SVD) of a set of Q^h representations taken from various sequences for Llama-3.1-8B. We find that the first right-vector of the SVD verifies Hypothesis A.1 for all tested heads, and we display examples of projection distributions in Figure 1b. The intuitive consequence of this observation regarding attention weights is that, if a given K_t^h has a strong projection along ϵu_h, then future queries $Q_{\geq t}^h$ can be expected to have a stronger dot-product with K_t^h in average.

However, it is not clear *a priori* that this effect is uni-directional, i.e. that there exists a unique direction u^h (up to a sign) that verifies Hypothesis A.1. Hence, identifying one such direction may not suffice to characterize the anisotropy of Q^h representations and to derive estimations of the dot-products used in attention. The uni-directional nature of the Query-Key anisotropy can be formalized as in Hypothesis A.2.

Hypothesis A.2. Let $u^h = \arg\max_{u \in \mathbb{S}^{d_H-1}} \mathbb{E}\left(\langle Q_i^h, u \rangle\right)$ and $B = (u^h, u_2, ..., u_{d_H})$ an orthonormal basis of $\mathbb{R}^{d_H}$. Then for all attention inputs X:

$$\forall m \in [2, d_H], \mathbb{E}\left(\langle Q_i^h, u_m \rangle\right) \approx 0$$

In Figure 1c, we observe that only the first singular component of the SVD of Q^h representations carries an anisotropic behavior, as the projections on all other components have a null mean. Hence, by taking the SVD right-vector basis as B, we can show that the first component of the SVD empirically verifies Hypothesis A.2.

B PROOF OF PROPOSITION 2.1

We begin the proof by writing $\langle Q_i^h, K_j^h \rangle$ in the basis B:

$$\mathbb{E}_{Q_i^h}(\langle Q_i^h, k \rangle) = \mathbb{E}_{Q_i^h}(\langle Q_i^h, u^h \rangle)\langle k, u^h \rangle + \sum_{m=2}^{d_h} \mathbb{E}_{Q_i^h}(\langle Q_i^h, u_m \rangle)\langle k, u_m \rangle$$

Hypothesis A.2 states that $\mathbb{E}_{i,X}(\langle Q_i^h, u_m \rangle) \approx 0$, which lets us do the following approximation:

$$\sum_{m=2}^{d_h} \mathbb{E}_{Q_i^h}(\langle Q_i^h, u_m \rangle)\langle k, u_m \rangle \approx 0$$

By combining Hypothesis A.1 and Hypothesis A.2, we also have that:

$$\mathbb{E}_{Q_i^h}(\langle Q_i^h, u^h \rangle) > 0$$

We conclude the proof by setting $\kappa^h = \mathbb{E}_{Q_i^h}(\langle Q_i^h, u^h \rangle)$.

Remark This result provides a justification for the method developed in Devoto et al. (2024). As a matter of fact, Hypothesis A.1 implies that $\mathbb{E}_j\left(\cos(K_j^h, u^h)\right)$ should have the same sign as ϵ. In practice, we observe $\epsilon = -1$ for a vast majority of heads in trained causal LMs. Hence, we can derive a looser estimation from Proposition 2.1:

$$\mathbb{E}_{i,X}(\langle Q_i^h, K_j^h \rangle) \approx -\kappa^h \left|\mathbb{E}_{j,X}\left(\cos(K_j^h, u^h)\right)\right| ||K_j^h||_2$$

This estimation shows that the L_2-norm of K_j^h vectors is negatively correlated with the corresponding mean attention logits and can therefore be used to approximate them. However, only using the L_2-norm to estimate the attention score as done in Devoto et al. (2024) is suboptimal, as it ignores the angular component of the $\langle K_j^h, u^h \rangle$ product.

C GENERATION RESULTS

We compute the final perplexity of Llama-3.1-70B in the memory-constrained setup for various compression factors and methods.

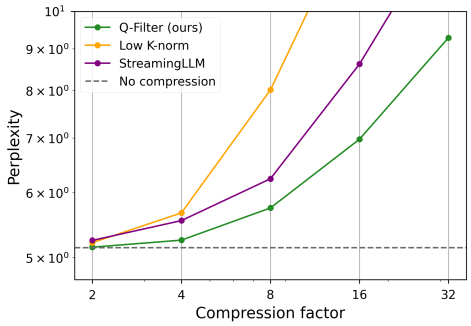
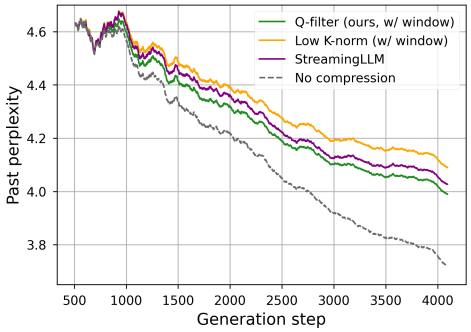
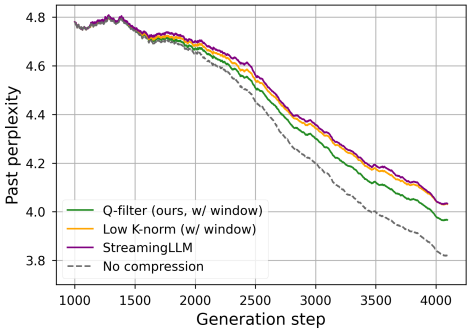


Figure 3: Final perplexity after 512 tokens for Llama-3.1-70B in the memory-constrained generation scenario.

We also run a study similar to the one conducted in Figure 6 with Qwen-2.5-7B-Instruct, which we display in Figure 4a, and with Llama-3.2-1B, which we display in Figure 4b.



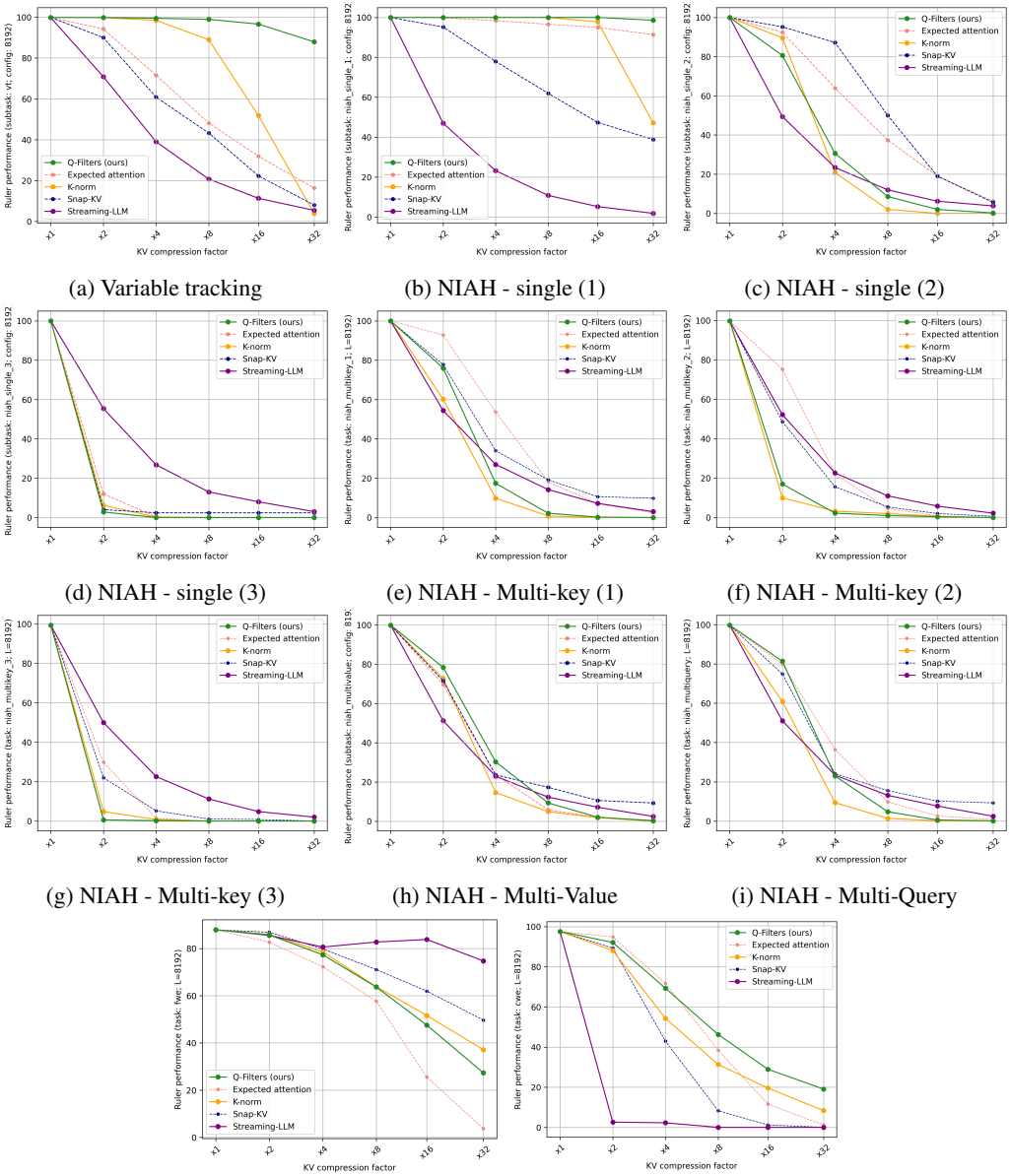
(a) Perplexity of the Qwen-2.5-7B-Instruct model along generation.



(b) Perplexity of the Llama-3.2-1B model along generation.

D RULER RESULTS

In Figure 5 we report detailed evaluation on the subsets of the Ruler dataset Hsieh et al. (2024).



(a) Variable tracking (b) NIAH - single (1) (c) NIAH - single (2)

(d) NIAH - single (3) (e) NIAH - Multi-key (1) (f) NIAH - Multi-key (2)

(g) NIAH - Multi-key (3) (h) NIAH - Multi-Value (i) NIAH - Multi-Query

(j) Frequent Words Extraction (FWE) (k) Common Words Extraction (CWE)

Figure 5: Performance of Llama-3.1-8B-Instruct using several KV Cache compression methods on individual tasks from the Ruler dataset (with length 8192) as compression ratio evolves. We report prompt compression methods using dotted lines for comparison.

E GENERATION EXAMPLES

Using Llama-3.1-8B, we identify interesting cases where Q-Filters provide the correct next token in a given long context, while K-norm and Streaming-LLM fail to capture the relevant information.

Text Sample (Context)	Q-Filters	K-Norm	Streaming-LLM
One of the show's first longest-running storylines was the rivalry between a young manicurist Jill Foster Abbott (Brenda Dickson, Jess Walton) and wealthy socialite, ***Katherine*** *Chancellor (****Jeanne*** *Cooper). [...] After much investigation, it is revealed that Kay is Jill's biological...*	*mother*	*father*	*father*
Both extreme right-wing leaders taught and practised the theology of ***Christian Identity****, a belief system which the FBI includes on its watch list as an extremist religion. [...] Here, the group trained an estimated 1,500 of like-minded Christian...*	*Identity*	*fundamental*	*fundamental*
The Viral Fever [...] ***TVF*** *debuted their platform, releasing the final two episodes of Pitchers on* ***TVF-Play****. [...]* ***TVF*** *claims to have worked with over 150 brands. [...] The show has been on hold as writer Biswapati Sarkar focuses on writing web series, including the sequel to TV...*	*F*	*_show*	*_show*

Table 2: Next-token generation examples for different KV Cache Compression methods, applied to Wikipedia article. Passages in bold correspond to useful information that is necessary to resolve the ambiguity in the choice of the next token.

F Implementation Details

For all our experiments, we use the popular Huggingface models with the recently released KVPress library Jegou & Jeblick (2024).

G Detailed Experimental Results

Language Modelling To evaluate the performance of Q-Filters in the language modelling setup, we perform generation on the Pile dataset Gao et al. (2020). We let the KV Cache grow up until a certain threshold, after which we start evicting the KV pairs so that the total size never exceeds the maximum threshold. We measure performance by tracking the model perplexity computed on past tokens in 20 sequences. We report results for a maximum KV Cache size of 512 pairs in Figure 6. We observe that Q-Filters consistently achieves the lowest perplexity among compression schemes, even for very long contexts. This observation scales to the 70B model, where Q-Filters significantly reduces the perplexity gap. This improvement is more pronounced in the latter portions of the sequences, suggesting better retention of relevant contextual information.

Needle in a Haystack The Needle-in-a-Haystack task embeds a key piece of information (the "needle") within a long sequence of distractors (the "haystack"), followed by a question that requires retrieving the needle. This evaluates the model's ability to handle long-range dependencies and tests how well KV Cache compression retains critical information. If important KV pairs are evicted, the model fails to answer correctly.

We evaluate Q-Filters by placing the needle at depths from 1k to 64k tokens and measuring retrieval accuracy. As shown in Figure 7, Q-Filters outperforms K-Norm (Devoto et al., 2024), preserving crucial information even in extremely long contexts.

Ruler Tasks We evaluate the proposed method on the Ruler dataset Hsieh et al. (2024), which comprises several sub-tasks that test the model long context modelling abilities, including Multi-hop Tracing, Long Context Aggregation, Long Context Retrieval and Question Answering. The dataset offers 3 variants with different sequence lengths: 4096, 8192, and 16384. We compare the score on Ruler with several other KV Cache compression methods and show average results in Figure 8a.

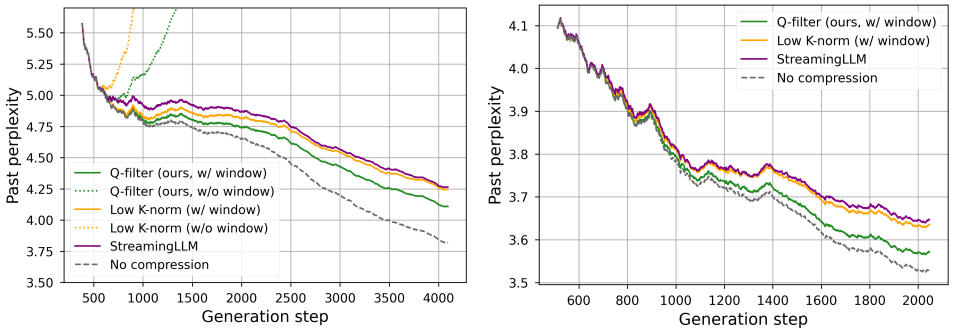
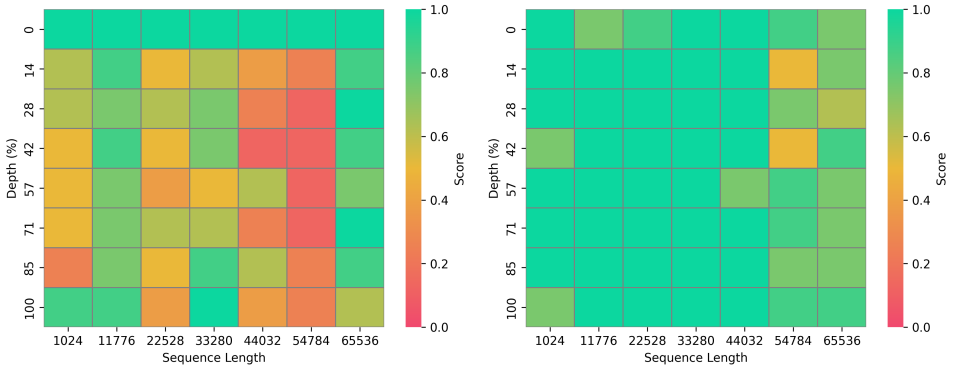


Figure 6: Generation performance for a KV Cache size limited to 512 items for Llama-3.1-8B (left) and Llama-3.1-70B (right).



(a) K-norm (average accuracy: 63%)

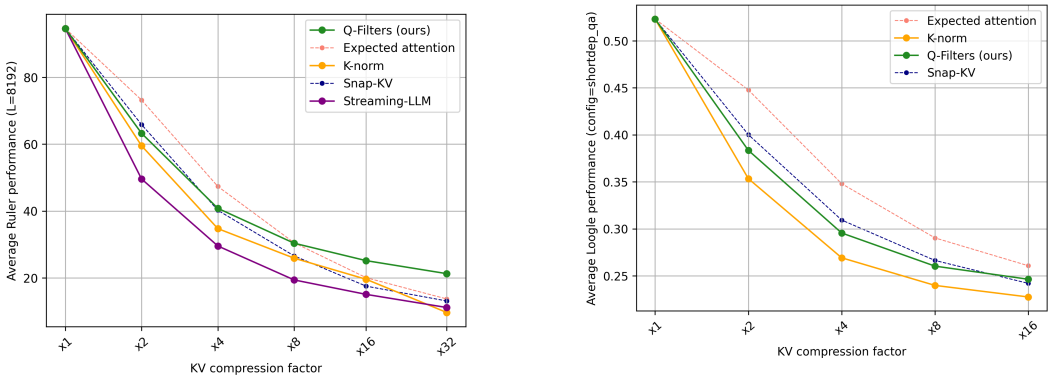
(b) Q-filters (average accuracy: 91%)

Figure 7: Needle-in-a-haystack performance for Llama-3.1-8B using 64x KV Cache compression.

We report detailed per-task results in Table 1 and in Appendix D. We test the model's score for several compression factors ranging from $2\times$ to $32\times$. While for some lower compression factors, we find performance on par with other methods, Q-Filters achieve the highest score with the strongest compression factor of $32\times$, demonstrating the method's effectiveness at high compression rates.

H Robustness of the Calibration Dataset

In Figure 9, we analyse how the calibration dataset size impacts the performance of our Q-Filters computation. Our experimental results demonstrate that increasing the number of samples in the calibration dataset leads to an improvement in average perplexity, although the marginal benefits diminish beyond a certain point, namely around 1k samples. This suggests that while larger calibration datasets generally produce more robust Q-Filters, there exists a practical trade-off balancing computational cost and performance benefits. Based on these empirical findings and computational efficiency considerations, we standardized our experimental protocol to utilize 3,000 samples for computing the Q-Filters across all subsequent experiments. Another important consideration in the development of robust Q-Filters is the choice of calibration dataset. To investigate this aspect, we conducted a systematic analysis using multiple diverse datasets and model versions in Figure 10. Our experiments revealed that the Q-Filter vectors exhibit remarkable stability across different calibration datasets, with a high average cosine similarity between vectors computed from distinct sources. This finding suggests that our method is relatively insensitive to the specific choice of calibration data, provided it maintains sufficient diversity and quality. Based on these results, we opted to use a carefully curated subset of the Pile dataset (Gao et al., 2020) for all Q-Filter computations.



(a) Average performance on Ruler (8192)

(b) Average performance on Loogle (Short Dependency QA)

Figure 8: Average score for different long-context benchmarks using Llama-3.1-8b with different methods and compression ratios

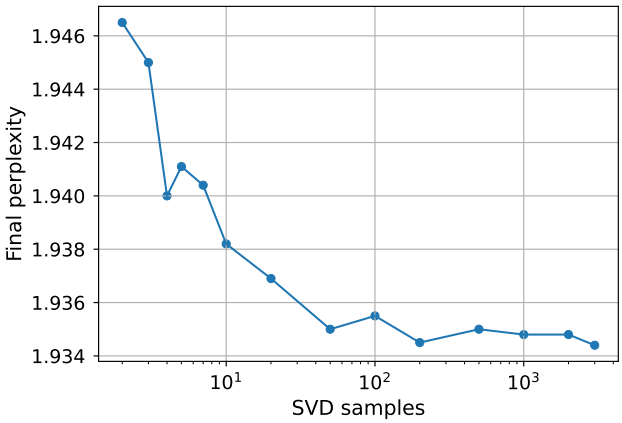


Figure 9: Perplexity after 1024 tokens for Q-Filters obtained using different counts of Q^h representations to calculate the SVD.

I Q-Filters Estimation Overhead

It could be argued that our method introduces a memory overhead as we need to store the Q-Filters on-device. Nevertheless, for a model using l layers and n_H heads, storing the Q-Filters requires $l \times n_H \times d_H$ parameters. For Llama-3.2-1B, this is 36k$\times$ smaller than the total parameter count and 196k$\times$ smaller in the case of Llama-3.2-405B. Another source of overhead could be attributed to the initial computation of the filters that are required for every new model. We find that passing 20 samples of length 2048 through the model and performing the SVD on 3k randomly sampled representations for each head is sufficient to obtain strong performance. In our experiments with Llama-3.2-70B, computing the filters took less than 3 minutes on two A100-80GB GPUs. This cost is thus negligible when compared with the cost of inference.

J Related Works

After the success of long-context models (Reid et al., 2024; Anthropic, 2024; Achiam et al., 2023), compressing the KV Cache has become a key research focus to enable processing of long-context inputs.

Some methods reduce the KV Cache size by modifying the model architecture. For example, Ainslie et al. (2023) and Shazeer (2019) reuse the same Keys for multiple queries, thereby reducing

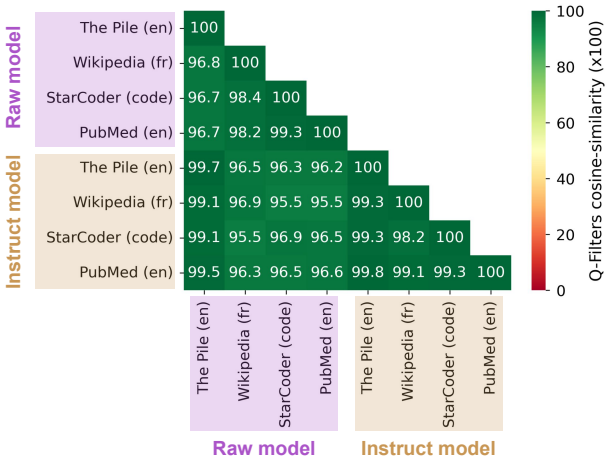
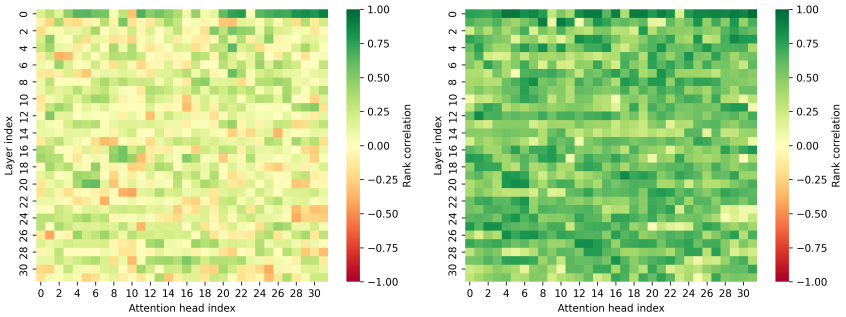


Figure 10: Cosine-similarity between Q-Filters computed on datasets coming from different domains and languages and on pre-trained and post-trained models. The scores are averaged over all layers and heads.



(a) K-norm (b) Q-Filter

Figure 11: Spearman rank correlation between KV compression scoring metrics and the observed attention S^h for Llama-3.2-1B, for K-norm (top) and Q-Filters (bottom).

redundancy in storage. Nawrot et al. (2024) propose a dynamic token-merging strategy, learning which KV pairs to merge. While these approaches achieve significant compression, they require training or fine-tuning, making them less practical in real-world scenarios where retraining the model from scratch is not feasible. In contrast, our method requires only a short, computationally inexpensive calibration step, avoiding parameter updates entirely. Recently DeepSeek-AI et al. (2024) introduced a Multi-Head Latent Attention, a modification to the standard attention mechanism that performs a low-rank reduction of the KV Cache during pre-training.

Training-free approaches aim to compress the KV Cache without modifying the model, typically by approximating the attention score over long sequences and prioritizing tokens with higher importance. Among these, Xiao et al. (2024) focus on language modelling tasks and propose always retaining the first token(s) (as an attention sink) and the last n tokens in a sliding window. Also, Zhang et al. (2024) focuses on generation tasks and introduces a policy that evicts tokens during generation based on a scoring function derived from cumulative attention. In contrast, other works focus on the task of compressing a large prompt provided by the user. Li et al. (2024) uses attention from the last part of the prompt to estimate KV pairs importance. With the same goal, Cai et al. (2024) assigns more cache budget to lower layers and less to higher layers. Finally, Guo et al. (2024) proposes to rescale the KV score of other methods by the L_1 norm of the Values.

In contrast, our approach is not tailored to a specific use case but provides competitive performance across both synthetic tasks and real-world scenarios, including in-context learning and chat-based

interactions. Additionally, many of these approaches are incompatible with FlashAttention Dao (2024) due to their reliance on accessing the full attention weights, which FlashAttention does not expose.

K LIMITATIONS

In Appendix C, we run generation experiments on Qwen-2.5-7B-Instruct (Qwen et al., 2025), and we observe that, although the results still favour the Q-Filters method, the gap is less clear compared to the Llama models. Our main hypothesis for this discrepancy lies in the slightly different attention mechanism used in Qwen-2.5 suite, which adds a bias to the QKV projection. Hence, it is likely that the geometrical observations made in Section 2 are not accurate in that case. Similarly, initial experiments with Olmo-2 models (OLMo et al., 2025) were unsuccessful, which can be explained by their use of the QK-normalization technique (Dehghani et al., 2023). These different tricks would most likely require an adaptation of our analysis to yield a better approximation of the attention distributions.