MagicPIG: LSH Sampling for Efficient LLM Generation

Zhuoming Chen[1] **Ranajoy Sadhukhan**[1] **Zihao Ye**[2] **Yang Zhou**[1]
Jianyu Zhang[3,4] **Niklas Nolte**[4] **Yuandong Tian**[4] **Matthijs Douze**[4]
Leon Bottou[4] **Zhihao Jia**[1] **Beidi Chen**[1,4]
[1]Carnegie Mellon University [2]University of Washington [3]New York University [4]FAIR, Meta
{zhuominc,rsadhukh,yangzho6,zhihaoj2,beidic}@andrew.cmu.edu
zhye@cs.washington.edu, jianyu@nyu.edu
{nolte,yuandong,matthijs,leonb}@meta.com

Abstract

Large language models (LLMs) with long context windows have gained significant attention. However, the KV cache, stored to avoid re-computation, now becomes a bottleneck. Leveraging the common insight that attention is sparse, various dynamic sparse or TopK-based attention approximation methods have been proposed. In this paper, we first show that TopK attention itself suffers from a quality degradation in certain downstream tasks because attention is not always as sparse as expected. Rather than selecting the keys and values with the highest attention scores, sampling with theoretical guarantees can provide a better estimation for attention output. To make the sampling-based approximation practical in LLM generation, we propose MagicPIG, a heterogeneous system based on Locality Sensitive Hashing (LSH). MagicPIG significantly reduces the workload of attention computation while preserving high accuracy for diverse tasks. MagicPIG stores the LSH hash tables and runs the attention computation on CPU, which allows to serve longer contexts and larger batch sizes with high approximation accuracy. MagicPIG can improve decoding throughput by $1.9 \sim 3.9\times$ across various GPU hardware and achieve 110ms decoding latency on a single RTX 4090 for Llama-3.1-8B-Instruct model with a context of 96k tokens.

1 Introduction

Large language models (LLMs) with long context windows, such as GPT [Achiam et al., 2023], Llama [Dubey et al., 2024], and Gemini [Team et al., 2023], have gained significant attention for their ability to enhance applications like chatbots [Chiang et al., 2024], search engines [Wang et al., 2024], and video analysis [Cheng et al., 2024]. However, serving long-context LLMs is highly challenging due to the unique bottleneck in auto-regressive generation—the key-value (KV) cache, which stores intermediate attention keys and values to avoid re-computation [Pope et al., 2022, Zhang et al., 2023]. Specifically, the KV cache grows linearly with both the batch size and sequence length, occupying substantial GPU memory and increasing decoding time. Moreover, the KV cache makes LLM generation extremely memory-bound, leading to underutilization of GPU computational power. For instance, an NVIDIA A100-40GB GPU can only handle a single request

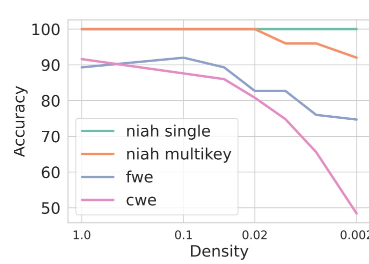
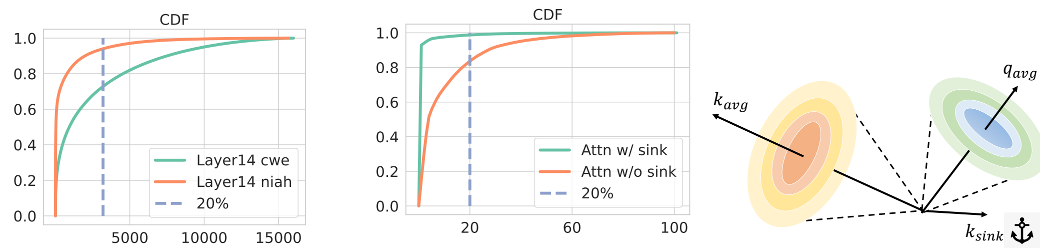


Figure 1: While TopK attention performs well on information retrieval tasks (niah) where the useful information reduces to a few words, it degrades severely in harder aggregated tasks like word extraction (cwe, fwe). x-axis: proportion of attention keys used for TopK attention.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).



(a) Long tailed phenomena **(b)** Attention sink reshapes sparsity **(c)** Geometry of attention

Figure 2: Left: Examples of long tailed distribution in LLM. The x-axis is the fraction (or number of tokens) used in the TopK, a.k.a. the *sampling budget*. **Mid:** Sink tokens make attention score look sparser. **Right:** The geometry of attention. The key states of attention sink k_0 is almost opposite to other tokens and its orientation is surprisingly invariant of input tokens. Query states lie close to k_0, thus forming attention sink and Figure 2b. k usually lie in a narrow cone that is far away from q. In certain heads, this geometry will result a long tailed distribution of attention score as well as the difficulty to search for the TopK keys.

for Llama with a 128k context length, with nearly half of the decoding time spent accessing the KV cache, and poor GPU utilization [He and Zhai, 2024].

Leveraging the common insight that attention is naturally sparse, dynamic sparse or TopK-based approximation has been extensively studied [Tang et al., 2024, Singhania et al., 2024, Zhang et al., 2024, Wu et al., 2024], but three major challenges prevent a wide adoption in LLM serving systems. (1) **Quality Degradation.** They usually propose various strategies to approximate a subset of KV cache that yields the highest attention scores. However, TopK attention itself is a biased attention approximation and lacks theoretical guarantees. Figure 1 shows that even exact TopK attention results significantly degrade the accuracy of certain downstream tasks. (2) **High Overhead.** There is a large overhead to identify TopK attention, which becomes the bottleneck rather than the attention computation. For example, as studied in Wu et al. [2024], naively applying search algorithms like IVF [Douze et al., 2024] requires access over 30% key states to obtain the exact TopK, showing an unsatisfying trade-off between search accuracy and cost. (3) **No Memory Saving.** Although saving KV cache loading time, they cannot reduce the total memory occupied by the KV cache, which limits the maximum context and batch sizes when VRAM is scarce.

An ideal sparse attention approximation approach should (1) preserve full accuracy for a diverse set of downstream tasks with guarantees, (2) involve low-cost overhead for KV cache selection, and (3) save GPU memory. The following observations, together with the performance drop shown in Figure 1 suggest that to achieve such demanding requirements, we need to go beyond TopK attention:

- *Attention is not always sparse*. Contradictory to previous belief [Zhang et al., 2023, 2024, Tang et al., 2024, Wu et al., 2024], we observe that attention is not always sparse, especially for tasks that leverage the full context. As shown in Figure 2a, in some layers, attention distribution can be very long-tailed, *i.e.*, the $\mathrm{Top}20\%$ attention can only cover 70% of the total attention scores.

- *Seemingly high sparsity is usually a consequence of an attention sink.* Most of the attention scores concentrate on initial tokens (attention sink phenomenon) [Xiao et al., 2023], making the distribution look sparser. However, as shown in Figure 2b, attention scores are distributed more uniformly among tokens except for the sink. According to the geometrical interpretation of sink, keys, and queries shown in Figure 2c, the attention sink, which we found surprisingly almost static regardless of the input token, is just for imposing sparsity on the attention distribution.

- *It is hard to find* TopK *attention.* Figure 2c also shows why searching for the Top-K keys is intrinsically costly. The keys and queries usually lie within two narrow cones with nearly opposite orientations, except for the attention sink. This significant mismatch between query and data distributions causes nearest-neighbor search methods to perform poorly.

These limitations of TopK attention requires rethinking the sparse attention approximation. Rather than only using the keys and values with highest scores, leveraging information on the distribution can make the estimation more accurate. We approach this as as bias correction problem in sampling. Unbiased and efficient sampling has been long studied in biology [Lukacs, 2009], sociology [Chen et al., 2018] as well as machine learning [Backurs et al., 2019, Chen et al., 2019, Zandieh et al., 2023], with theoretical guarantees.

Figure 3 shows that sampling values according to their corresponding attention score (we call this *oracle sampling*) achieves a much lower (up to $4\times$) estimation error than the naive TopK selection. Deploying sampling estimation in attention is promising, but three challenges remain. First, how a reduction of the attention error can make a difference in downstream performance is unclear [Backurs et al., 2019, 2018]. Second, modeling the attention scores distribution is necessary for efficient sampling, but inferring the distribution parameters requires expensive computations. Third, fully leveraging the resources of modern hardware, GPU and CPU, with a theoretically efficient algorithm is non-trivial.

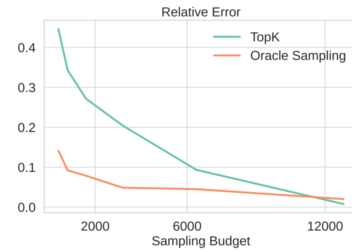


Figure 3: TopK v.s. Sampling, 16k total context

In this paper, we propose Magic samPlIng for Generation (MAGICPIG), which leverages Locality sensitive hashing (LSH) sampling for efficient LLM generation. LSH is employed for sampling to approximate the attention scores distribution and estimate attention output. By computing hash functions on GPU and conducting sampling on CPU, MAGICPIG can allow massive hash tables and hash functions compared to prior work [Kitaev et al., 2020, Chen et al., 2021], which are of vital importance for accurate estimation [Backurs et al., 2018]. Following the practice of Aminabadi et al. [2022], He and Zhai [2024], we offload the KV cache computation, which is memory bound, to CPU to allow a larger batch or longer context.

2 MAGICPIG

Figure 3 (more details in Appendix B) demonstrates the potential of sampling-based estimation. In Section 2.1, we show the practical algorithm. In Section 2.2, we demonstrate our system co-design for accurate and efficient LLM decoding through GPU-CPU collaboration.

Note that most of the derivations in this section might be classical and can even be found in textbooks. Still, our goal is to leverage them to motivate MAGICPIG design and precisely demonstrate the power of a rigorously sound algorithm with system co-design in deep generative models.

2.1 Algorithm implementation

To make estimation via LSH sampling practical (explained in Appendices C.1 and C.2), MAGICPIG is implemented by the following design.

Estimator approximation.

The probabilities provided by hashing are not normalized. Hence we adapt our estimator: After obtaining S with probability u, MAGICPIG computes

$$X = \frac{\sum_{i=1}^{n} \frac{\widetilde{w_i}}{u_i} v_i \mathbf{1}_{i\in S}}{\sum_{i=1}^{n} \frac{\widetilde{w_i}}{u_i} \mathbf{1}_{i\in S}} = \frac{\sum_{i\in S} \frac{\widetilde{w_i}}{u_i} v_i}{\sum_{i\in S} \frac{\widetilde{w_i}}{u_i}} \qquad (1)$$

where $\widetilde{w_i} = e^{\frac{qk_i^T}{\sqrt{d}}}$ and u_i is the probability of k_i is sampled by hashing.

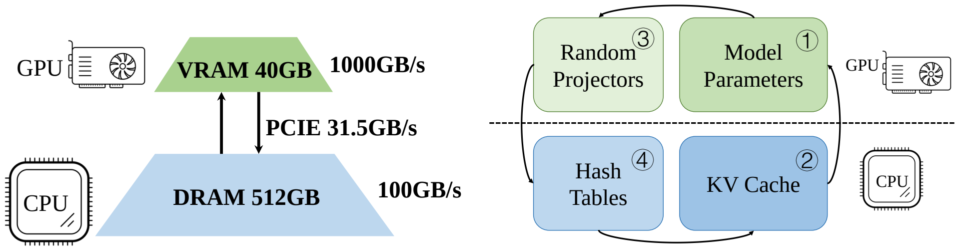
Hash function selection. MAGICPIG leverages **SimHash** [Sadowski, 2007], that draws with $K \times L$ random vectors. For each of the L hash tables, the q and k_is vectors are projected on K directions and only the sign of the projection is kept, which yields a K-bit hash value. Key k_i is sampled only if there exist at least **two** hash tables where k_i shares the hash value with q. The corresponding probability is

$$u_i = \mathbb{P}[k_i \text{ is sampled}] = 1-(1-p^K)^L - Lp^K(1-p^K)^{L-1} \quad \text{where} \quad p = 1-\frac{1}{\pi}\arccos\frac{qk_i{}^T}{|q|\cdot|k_i|} \qquad (2)$$

Algorithm 1: MAGICPIG Decoding

Input: $\boldsymbol{K},\boldsymbol{V} \in R^{n\times d}$, $\boldsymbol{q} \in R^{1\times d}$, random projectors $\boldsymbol{W} \in R^{d\times(K\times L)}$, hash tables $\boldsymbol{HT}$.
Compute hash code for new query
$\boldsymbol{q}_{\text{code}} = \textbf{Encode}(\boldsymbol{q},\boldsymbol{W})$
Query hash tables to sample $\boldsymbol{S}$ in Equation (1)
$\boldsymbol{S} = \textbf{Query}(\boldsymbol{HT},\boldsymbol{q}_{\text{code}}), \boldsymbol{K_S} = \boldsymbol{K}[\boldsymbol{S}], \boldsymbol{V_S} = \boldsymbol{V}[\boldsymbol{S}]$
Compute inner product for $\boldsymbol{q}$ and sampled $\boldsymbol{K}$
$\boldsymbol{w} = \boldsymbol{q}\boldsymbol{K_S^T}$
Compute collision probability for each hash function
$\boldsymbol{p} = 1-\boldsymbol{w}/(||\boldsymbol{q}||\cdot||\boldsymbol{K_S}||)$
Compute sampling probability
$\boldsymbol{u} = 1-(1-\boldsymbol{p}^K)^L - L\boldsymbol{p}^K(1-\boldsymbol{p}^K)^{L-1}$
Compute attention output estimation
$\bar{\boldsymbol{o}} = \textbf{Softmax}(\frac{\boldsymbol{w}}{\sqrt{d}} - \log(\boldsymbol{u}))\boldsymbol{V_S}$
Return $\bar{\boldsymbol{o}}$

Data pre-processing. Before building hash tables, MAGICPIG centers the k_i vectors. As shown in Figure 2c, keys are almost always concentrated on one side of the queries, except the initial token. Random projections cannot effectively distinguish keys in this case, resulting in uniform sampled



(a) Memory Hierarchy **(b)** Workload Partition

Figure 4: Left: Memory hierarchy of hardware. GPU VRAM has high bandwidth but is limited. CPU DRAM is sufficient but is relatively slow. The limited bandwidth of PCIE forbids large-scale data transferring. **Right:** Workload partition of MAGICPIG. Linear projections and hash function computation (by random projection) are done on GPU, while sampling with hash tables and attention are done CPU. The execution order is ①③④②.

probabilities. Luckily, Softmax is translation invariant. Centering ($\bar{k_i} = k_i - \frac{1}{n}\sum_{i=1}^{n} k_i$) distributed the keys better and remains computationally equivalent.

Our algorithm applies to a single attention head, see Algorithm 1. The details of **Encode**, **Query** as well as the hash table construction are described in prior work [Sadowski, 2007, Chen et al., 2020b].

2.2 System co-design

The memory size of KV cache remains a bottleneck for long-context LLM decoding, especially when GPU VRAM is limited. DRAM on the CPU side offers sufficient memory capacity with $100-200$GB/s bandwidth, which is usually $10-20\%$ of GPU VRAM bandwidth (see Figure 4a). Ideally, this gap can be mitigated by $5-10\times$ sparsity. To make CPU DRAM an *aggregated memory* for GPU, the workload must be partitioned. In our experiments $K=9$ or 10 and L is a few hundreds.

Our system design extends prior work [He and Zhai, 2024, Aminabadi et al., 2022] by spliting LLM decoding into three parts. (1) Parameter computations, ie. all linear projectors including MLP and W_Q, W_K, W_V, W_O in the self-attention module runs on GPU. (2) Attention computation, which involves $o = \text{Softmax}(\frac{qK^T}{\sqrt{d}})V$, runs on CPU. (3) Random projections. At generation time, for each q, $K \times L$ random projections are conducted to obtain the hash codes. Since all heads can share the same random projectors, the memory overhead is limited (25 MB in our implementation), so this step is compute-bound. Therefore, the projection is placed on GPU. (4) Retrieval. The hash codes of q, need to be looked up in L hash tables, which is negligible computationally. However, the pre-built hash tables for k_is can occupy considerable memory, making it a better fit for CPU. With the above partition, we are able to support hash tables with K and L beyond the scale of prior work [Kitaev et al., 2020, Chen et al., 2021, Zandieh et al., 2023] without worrying about computation for hash codes as well as the storage of hash tables.

3 Evaluation

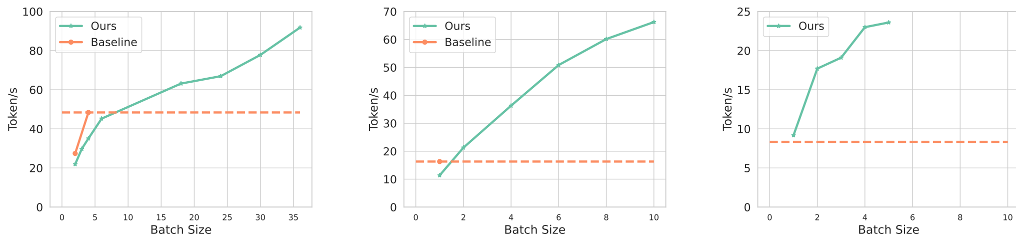
In this section, we aim to demonstrate that MAGICPIG can speed up LLM decoding while preserving high accuracy. We first present MAGICPIG's accuracy in downstream tasks, followed by our end-to-end system results showing wall-clock performance.

3.1 MAGICPIG Preserves Accuracy

We demonstrate MAGICPIG can preserve accuracy in diverse tasks with less than 5% computation.

Setup. Our experiments are based on Llama [AI@Meta, 2024, Dubey et al., 2024, Touvron et al., 2023] models. RULER [Hsieh et al., 2024] is evaluated with 50 examples per task. Other evaluations on lm-eval-harness [Gao et al., 2021] and LongBench [Bai et al., 2023] are in Tables 2 and 3.

Baselines. Besides full attention, Quest [Tang et al., 2024] and its variants are used as baselines. In its default setting, Quest uses a "page size" of 16, which is 1/16 of the full attention cost. To compare the methods fairly in the low computation budget regime, we also evaluate Quest with page size 32 and 64 and ensure that at least one page is selected in every test example. The initial 4 tokens and local 64 (for LongBench [Bai et al., 2023] and RULER [Hsieh et al., 2024]) or 24 (for lm-eval-harness [Gao et al., 2021]) tokens as well as layer-$\{0,16\}$ are statically preserved. We do not use the theoretical transformations in our implementations in Equation (10) as we do not find them to contribute to accuracy improvements.



(a) A100 with 34B model **(b)** L40 with 13B model **(c)** RTX 4090 with 8B model

Figure 5: We evaluate MAGICPIG on three serving scenarios. **Left:** A100 serves 34B model with 16K context. MAGICPIG achieves $1.9\times$ throughput improvement. **Mid:** L40 serves 13B model with 16K context. MAGICPIG achieves $3.9\times$ throughput improvement. **Right:** Simulated RTX 4090 serves 8B model with 128K context. MAGICPIG achieves a latency of 110ms in single request serving and can improve the throughput of baseline by up to $2.9\times$. The dash lines denote the highest throughput of baselines. With KV cache offloading, MAGICPIG can fit much larger batch size compared with full attention on GPU, which contributes to the throughput improvement.

Cost. The cost for the attention approximation consists of two parts: Cost_1 is the sampling/search cost, Cost_2 is the attention computation cost. We report the ratio of number of FLOPs compared of the full attention computation. For MAGICPIG, $\text{Cost}_1 \simeq 0$ and Cost_2 is empirically measured for different LSH hyper-parameters. For Quest with page size K, $\text{Cost}_1 = \frac{1}{K}$ and Cost_2 is controlled manually.

Analysis. From Tables 1, 2 and 3, (1) MAGICPIG preserves high accuracy (degradation less than 2%) with a computation cost of $2\% \sim 5\%$. (2) With LSH sampling which introduces an order of magnitude lower sampling/searching cost (Cost_1), MAGICPIG can achieve equivalent or better accuracy with only half of the computation cost.

Table 1: Synthesized tasks on RULER [Hsieh et al., 2024]. MAGICPIG preserves high accuracy with low computation. Config and cost are defined as in Table 2.

Methods	Config	16K	32K	64K	96K	Avg.	Cost_1	Cost_2	Cost_{total}.
Llama-3.1-8B-Instruct	Full	94.2	91.5	86.1	83.0	88.7	0.00	1.00	1.00
MAGICPIG	(10,150)	91.8	88.9	84.8	80.0	86.4	0.00	0.02	0.02
MAGICPIG	(9,120)	93.4	90.6	84.7	81.5	**87.6**	0.00	0.04	0.04
MAGICPIG	(8,75)	92.9	90.2	84.9	81.7	87.4	0.00	0.05	0.05
Quest	(16,0.04)	86.3	85.4	81.9	74.9	82.1	0.06	0.05	0.11
Quest	(32,0.06)	84.3	84.0	80.1	74.4	80.7	0.03	0.06	0.09
Quest	(64,0.08)	85.2	84.3	77.0	74.2	80.2	0.02	0.08	0.10

3.2 MAGICPIG Shows Impressive Efficiency across Various Hardware Settings

We show MAGICPIG can bring up to $3.9\times$ wall clock speed up and reduce GPU memory consumption on different models and hardware settings (A100, L40, RTX4090).

Setup. We evaluate our system performance on 3 serving settings. (1) 80GB GPU (A100) and 34B model (CodeLlama-34B) [Rozière et al., 2024] with 16K contexts; (2) 48GB GPU (L40) and 13B model (CodeLlama-13B) [Rozière et al., 2024] with 16K contexts; (3) 24GB GPU (e.g. RTX 4090) and 8B model (Llama-3.1-8B) [Dubey et al., 2024] with 96K contexts.

Baselines. Our baselines for (1) and (2) are full attention on GPU and for (3) is full attention on CPU with theoretical estimated bandwidth. Our system's GPU part is implemented in native Pytorch [Paszke et al., 2019] and the CPU part in FBGEMM [Khudia et al., 2021] in bfloat16 precision.

Analysis. In Figures 5a to 5c, we demonstrate (1) MAGICPIG significantly improves decoding throughput for all three scenarios (A100: $1.9\times$, L40: $3.9\times$, RTX 4090: $2.9\times$) and can achieve a latency of 110ms for single request generation with 96K context for RTX 4090. (2) With KV cache offloading, MAGICPIG can fit much larger batches than GPU full attention baselines ($15 \sim 18\times$).

4 Conclusion

In this work, we first present the limitation of TopK attention approximation for addressing the computational and memory challenges of long-context LLM generation. Then we show oracle sampling can go beyond TopK and introduce MAGICPIG, a novel approach that leverages LSH sampling to approximate the oracle sampling. MAGICPIG significantly reduces the workload of attention computation while preserving high accuracy across diverse tasks. The theoretical soundness, robustness, and scalability of MAGICPIG open up new opportunities in both attention approximation methods and algorithm-hardware co-design.

References

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

Muhammad Adnan, Akhil Arunkumar, Gaurav Jain, Prashant Nair, Ilya Soloveychik, and Purushotham Kamath. Keyformer: Kv cache reduction through key tokens selection for efficient generative inference. *Proceedings of Machine Learning and Systems*, 6:114–127, 2024.

AI@Meta. Llama 3 model card. 2024. URL `https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md`.

Josh Alman and Zhao Song. Fast attention requires bounded entries. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 63117–63135. Curran Associates, Inc., 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/file/c72861451d6fa9dfa64831102b9bb71a-Paper-Conference.pdf`.

Reza Yazdani Aminabadi, Samyam Rajbhandari, Minjia Zhang, Ammar Ahmad Awan, Cheng Li, Du Li, Elton Zheng, Jeff Rasley, Shaden Smith, Olatunji Ruwase, et al. Deepspeed inference: Enabling efficient inference of transformer models at unprecedented scale. *arXiv preprint arXiv:2207.00032*, 2022.

Arturs Backurs, Moses Charikar, Piotr Indyk, and Paris Siminelakis. Efficient density evaluation for smooth kernels. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 615–626, 2018. doi: 10.1109/FOCS.2018.00065.

Arturs Backurs, Piotr Indyk, and Tal Wagner. Space and time efficient kernel density estimation in high dimensions. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper_files/paper/2019/file/a2ce8f1706e52936dfad516c23904e3e-Paper.pdf`.

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.

Jan van den Brand, Zhao Song, and Tianyi Zhou. Algorithm and hardness for dynamic attention maintenance in large language models. *arXiv preprint arXiv:2304.02207*, 2023.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2019. URL `https://arxiv.org/abs/1809.11096`.

Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02, page 380–388, New York, NY, USA, 2002. Association for Computing Machinery. ISBN 1581134959. doi: 10.1145/509907.509965. URL `https://doi.org/10.1145/509907.509965`.

Beidi Chen, Anshumali Shrivastava, and Rebecca C Steorts. Unique entity estimation with application to the syrian conflict. *The Annals of Applied Statistics*, 12(2):1039–1067, 2018.

Beidi Chen, Yingchen Xu, and Anshumali Shrivastava. Fast and accurate stochastic gradient estimation. *Advances in Neural Information Processing Systems*, 32, 2019.

Beidi Chen, Tharun Medini, James Farwell, sameh gobriel, Charlie Tai, and Anshumali Shrivastava. Slide : In defense of smart algorithms over hardware acceleration for large-scale deep learning systems. In I. Dhillon, D. Papailiopoulos, and V. Sze, editors, *Proceedings of Machine Learning and Systems*, volume 2, pages 291–306, 2020a. URL `https://proceedings.mlsys.org/paper_files/paper/2020/file/ca3480d82599b9b9b7040655483825c1-Paper.pdf`.

Beidi Chen, Tharun Medini, James Farwell, Charlie Tai, Anshumali Shrivastava, et al. SLIDE: In defense of smart algorithms over hardware acceleration for large-scale deep learning systems. *Proceedings of Machine Learning and Systems*, 2:291–306, 2020b.

Beidi Chen, Tri Dao, Eric Winsor, Zhao Song, Atri Rudra, and Christopher Ré. Scatterbrain: Unifying sparse and low-rank attention. *Advances in Neural Information Processing Systems*, 34:17413–17426, 2021.

Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, et al. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms. *arXiv preprint arXiv:2406.07476*, 2024.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *CoRR*, abs/2307.08691, 2023. doi: 10.48550/ARXIV.2307.08691. URL `https://doi.org/10.48550/arXiv.2307.08691`.

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

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022b.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*, 2021.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. *arXiv preprint arXiv:2401.08281*, 2024.

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

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, September 2021. URL `https://doi.org/10.5281/zenodo.5371628`.

Jiaao He and Jidong Zhai. Fastdecode: High-throughput gpu-efficient llm serving using heterogeneous pipelines. *arXiv preprint arXiv:2403.11421*, 2024.

Pujiang He, Shan Zhou, Wenhuan Huang, Changqing Li, Duyi Wang, Bin Guo, Chen Meng, Sheng Gui, Weifei Yu, and Yi Xie. Inference performance optimization for large language models on cpus, 2024. URL `https://arxiv.org/abs/2407.07304`.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Timothy Hesterberg. Advances in importance sampling. 01 2003.

Ke Hong, Guohao Dai, Jiaming Xu, Qiuli Mao, Xiuhong Li, Jun Liu, Kangdi Chen, Yuhan Dong, and Yu Wang. Flashdecoding++: Faster large language model inference on gpus, 2024. URL `https://arxiv.org/abs/2311.01282`.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*, 2001. URL `https://www.aclweb.org/anthology/H01-1069`.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Omid Jafari, Preeti Maurya, Parth Nagarkar, Khandker Mushfiqul Islam, and Chidambaram Crushev. A survey on locality sensitive hashing algorithms and their applications. *arXiv preprint arXiv:2102.08942*, 2021.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017. URL `https://arxiv.org/abs/1705.03551`.

Daya Khudia, Jianyu Huang, Protonu Basu, Summer Deng, Haixin Liu, Jongsoo Park, and Mikhail Smelyanskiy. Fbgemm: Enabling high-performance low-precision deep learning inference. *arXiv preprint arXiv:2101.05615*, 2021.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

Teun Kloek and Herman K Van Dijk. Bayesian estimates of equation system parameters: an application of integration by monte carlo. *Econometrica: Journal of the Econometric Society*, pages 1–19, 1978.

Xin Li and Dan Roth. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002. URL `https://www.aclweb.org/anthology/C02-1150`.

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.

Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. Qserve: W4a8kv4 quantization and system co-design for efficient llm serving. *arXiv preprint arXiv:2405.04532*, 2024.

Tianyang Liu, Canwen Xu, and Julian McAuley. Repobench: Benchmarking repository-level code auto-completion systems, 2023. URL `https://arxiv.org/abs/2306.03091`.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024.

Sharon L Lohr. *Sampling: design and analysis*. Chapman and Hall/CRC, 2021.

Paul Lukacs. Closed population capture-recapture models. *Program MARK: a gentle introduction*, 8, 2009.

Yuzhen Mao, Martin Ester, and Ke Li. Iceformer: Accelerated inference with long-sequence transformers on cpus. *arXiv preprint arXiv:2405.02842*, 2024.

Art B. Owen. *Monte Carlo theory, methods and examples*. `https://artowen.su.domains/mc/`, 2013.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Anselm Levskaya, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *arXiv preprint arXiv:2211.05102*, 2022.

Siva Reddy, Danqi Chen, and Christopher D Manning. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266, 2019.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code, 2024. URL `https://arxiv.org/abs/2308.12950`.

Caitlin Sadowski. Simhash : Hash-based similarity detection. 2007. URL `https://api.semanticscholar.org/CorpusID:199497165`.

Anshumali Shrivastava and Ping Li. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2321–2329, 2014.

Prajwal Singhania, Siddharth Singh, Shwai He, Soheil Feizi, and Abhinav Bhatele. Loki: Low-rank keys for efficient sparse attention. *arXiv preprint arXiv:2406.02542*, 2024.

Ryan Spring and Anshumali Shrivastava. A new unbiased and efficient class of lsh-based samplers and estimators for partition function computation in log-linear models. *arXiv preprint arXiv:1703.05160*, 2017.

Hanshi Sun, Zhuoming Chen, Xinyu Yang, Yuandong Tian, and Beidi Chen. Triforce: Lossless acceleration of long sequence generation with hierarchical speculative decoding. *arXiv preprint arXiv:2404.11912*, 2024.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference. *arXiv preprint arXiv:2406.10774*, 2024.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

Minzheng Wang, Longze Chen, Cheng Fu, Shengyi Liao, Xinghua Zhang, Bingli Wu, Haiyang Yu, Nan Xu, Lei Zhang, Run Luo, et al. Leave no document behind: Benchmarking long-context llms with extended multi-doc qa. *arXiv preprint arXiv:2406.17419*, 2024.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval head mechanistically explains long-context factuality. *arXiv preprint arXiv:2404.15574*, 2024.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

Zihao Ye, Ruihang Lai, Bo-Ru Lu, Chien-Yu Lin, Size Zheng, Lequn Chen, Tianqi Chen, and Luis Ceze. Cascade inference: Memory bandwidth efficient shared prefix batch decoding, February 2024. URL `https://flashinfer.ai/2024/02/02/cascade-inference.html`.

Amir Zandieh, Insu Han, Majid Daliri, and Amin Karbasi. Kdeformer: Accelerating transformers via kernel density estimation. In *International Conference on Machine Learning*, pages 40605–40623. PMLR, 2023.

Hailin Zhang, Xiaodong Ji, Yilin Chen, Fangcheng Fu, Xupeng Miao, Xiaonan Nie, Weipeng Chen, and Bin Cui. Pqcache: Product quantization-based kvcache for long context llm inference. *arXiv preprint arXiv:2407.12820*, 2024.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang "Atlas" Wang, and Beidi Chen. H2o: Heavy-hitter oracle for efficient generative inference of large language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 34661–34710. Curran Associates, Inc., 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/file/6ceefa7b15572587b78ecfcebb2827f8-Paper-Conference.pdf`.

A Background

In this section, we formulate the targeted attention estimation problem and related works.

A.1 Problem formulation

In LLM decoding phase, self attention part calculates a weighted average of previous values by

$$o = \mathrm{Softmax}(\frac{qK^T}{\sqrt{d}})V = wV \quad q \in \mathbb{R}^{1\times d} \quad K,V \in \mathbb{R}^{n\times d} \quad w \in \mathbb{R}^{1\times n} \tag{3}$$

where d is the head dimension and n is the context size. $K = [k_1,k_2,...,k_n], V = [v_1,v_2,...,v_n], k_i,v_i \in \mathbb{R}^{1\times d}$ is KV cache. Normalized attention weight $w = \mathrm{Softmax}(\frac{qK^T}{\sqrt{d}}) \in \mathbb{R}^{1\times n}$ is also called attention (score) distribution. Our target is to find sampling matrix $\Pi \in \mathbb{R}^{n\times m}$ and diagonal matrix $D \in \mathbb{R}^{m\times m}$ which minimize

$$\delta = ||wV - w\Pi D \Pi^T V|| \tag{4}$$

where $m \ll n$ is computation budget. For TopK attention, suppose $w_{r_1} > ... > w_{r_m} > ... > w_{r_n}$, then

$$\Pi_{i,j} = \begin{cases} 1, & \text{if } i = r_j, \\ 0, & \text{otherwise.} \end{cases} \quad D_{ii} = \frac{1}{\sum_{i=1}^{m} w_{r_i}} \tag{5}$$

A.2 Related works

Efficient Attention. Attention approximation has been long studied. Reformer [Kitaev et al., 2020], KDEformer [Zandieh et al., 2023] and ScatterBrain [Chen et al., 2021] tackle the problem via locality-sensitive hashing. These methods work in training and encoder models like BigGAN [Brock et al., 2019]. Theoretically, the error bounds and minimal workload required are continuously improved [Brand et al., 2023, Alman and Song, 2023] but not proven to be practical for wall-clock acceleration in LLM decoding. Besides, flash-attention [Dao et al., 2022b, Dao, 2023, Dao et al., 2022a], flash-decoding [Ye et al., 2024, Hong et al., 2024] and SlimAttention [He et al., 2024] losslessly accelerate scaled product attention operator by maximizing the utilization of hardware, which is orthogonal to our approach.

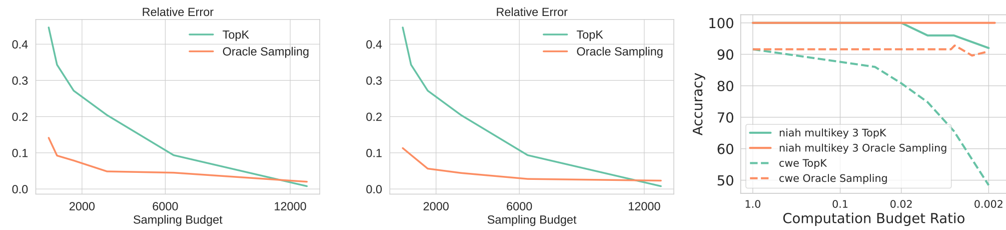
Locality sensitive hashing. Locality sensitive hashing (LSH) [Backurs et al., 2019, 2018] is a family of hashing functions which assigns the same hash codes for similar inputs with higher probability than others [Chen et al., 2020b, Jafari et al., 2021]. LSH uses two hyper-parameters, (K,L). L hash tables are independently built. Each hash table has its own function H, which projects a high-dimension vector to an integer by concatenating K random independent hash functions. In the sampling process, all vectors that share hash codes in at least one hash table with a query will be collected. **SimHash** [Charikar, 2002] is the LSH family based on cosine similarity. For a vector $x \in \mathbb{R}^d$, SimHash generates a random hyperplane w and returns $\mathrm{Sign}(w^T x)$. Vectors share the same sign if and only if the random projection is not in-between them. For a random projection, all angles are equally likely, thus the probability that two vectors x, y share the same sign for is $p = 1 - \frac{\theta}{\pi}$, where $\theta = \arccos \frac{xy^T}{||x||\cdot||y||}$. If we have L hash tables each with K random hash functions, the probability of y to be retrieved by query x is $1-(1-p^K)^L$.

KV Cache reduction. To get rid of memory bound introduced by KV cache thus enabling a larger batch size or serving a longer prompt, many methods are proposed to reduce the volume of KV cache. For example, H_2O [Zhang et al., 2023], SnapKV [Li et al., 2024] and Keyformer [Adnan et al., 2024] calculates heuristics during prefilling phase to decide which tokens to preserve for decoding phase. Quest [Tang et al., 2024] and Loki [Singhania et al., 2024] do not evict KV cache but apply dynamic sparsity to reduce KV Cache loading at inference time. Besides the reduction along the dimension of sequence length, methods like KIVI [Liu et al., 2024] and QServe [Lin et al., 2024] reduce the size of KV Cache by quantization.

B Rethinking attention sparsity

In this section, we examine TopK attention, which is the theoretical upper bound of prior search-based algorithms including both static methods [Zhang et al., 2023, Li et al., 2024] and dynamic methods [Tang et al., 2024, Singhania et al., 2024, Mao et al., 2024]. We show that TopK is *sub-optimal* and present another attention approximation based on sampling and estimation with an oracle, that improves the accuracy and/or the computation cost.

B.1 Achilles' heel of TopK attention



(a) Estimation error Layer11 **(b)** Estimation error Layer19 **(c)** Downstream Comparison

Figure 7: Left and Middle: Oracle sampling estimation can significantly reduce numerical error compared to TopK attention. The evaluated context size is 16k. The x-axis is *sampling budget* for oracle sampling and *computation budget* for TopK attention. Notice that the estimation error of TopK attention will cross oracle sampling after a certain large budget (12k in figures). This is because oracle sampling will repetitively sample the same subset of tokens with a high probability while TopK will not. Theorem B.3 further explains this. **Right:** Downstream comparison for oracle sampling estimation and TopK attention. The x-axis for both methods is *computation budget ratio*, i.e. the fraction of selected/sampled tokens.

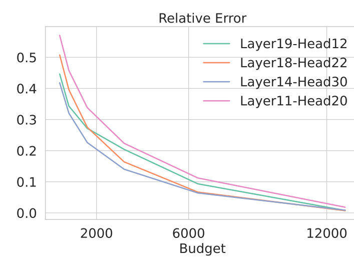


Figure 6: TopK estimation error for a KV-cache of 16k tokens.

As it is defined, TopK attention only computes the weighted average on elements with highest attention scores. To quantify its performance, the *computation budget* of TopK attention, is defined as the number of selected tokens, i.e. the K of TopK. Searching-based sparse attention algorithms, like [Tang et al., 2024, Singhania et al., 2024, Wu et al., 2024] are approximations for TopK attention by replacing the true TopK keys with the ones found by approximate searching algorithms.

However, we do find a significant performance degradation in downstream tasks caused by TopK attention as shown in Figure 1. Although TopK attention preserves accuracy for retrieval tasks that only require a minimal subset of the context (needle-in-a-haystack single/multikey [Hsieh et al., 2024]), it severely degrades for aggregation tasks that leverage the full context (common word extraction and frequent word extraction [Hsieh et al., 2024]). Intuitively, the information is distributed more broadly for aggregation tasks, which results in less peak attention score distribution.

TopK attention is *biased* and *inaccurate* especially when the distribution of attention scores is long-tailed, and the computation budget or density (i.e., K) is limited. Unfortunately, long tail phenomena do occur in LLMs across all layers (prior works [Xiao et al., 2023, Tang et al., 2024, Sun et al., 2024] usually skip the first two layers to maintain accuracy) as presented in Figure 2a. $\mathrm{Top}20\%$ tokens can only cover $70\sim80\%$ attention scores, leaving a large proportion of keys and values not considered, which is translated into a non-negligible $(15\sim20\%)$ estimation error in Figure 6.

B.2 Estimate attention with sampling

Existing TopK attention mechanisms ignore tokens in the KV cache with low attention scores, which introduces a bias since the ignored tokens sum up to a large proportion of attention scores (Figure 2a). As a result, TopK attention achieves suboptimal performance for long-context tasks, such as information aggregation (Figure 1). Increasing the computation budget for TopK attention does help reduce the estimation error (Figure 6) since it will involve more elements in computing; however, the following question is posed:

Can we improve the estimation quality with low computational budgets?

Inspired by *mark and recapture* [Lukacs, 2009, Owen, 2013, Lohr, 2021, Chen et al., 2018], we show in the following that attention output can be estimated with sampling. Using notations from Appendix A.1 we can re-write attention output o as the expectation of $v_i, 1 \le i \le n$ from distribution w, i.e. $o = \mathbb{E}_{i\sim w}(v_i)$, which can be estimated by the following method.

Definition B.1 (Oracle Sampling Estimation)**.** Given a sampling budget $\mathcal{B}$ and normalized attention score w, $\mathcal{B}$ elements are sampled independently from w (i.e. $i_1, i_2, ..., i_{\mathcal{B}} \overset{\text{iid}}{\sim} w$). Then the attention output is estimated as

$$\bar{o}=\frac{1}{\mathcal{B}}\sum_{j=1}^{\mathcal{B}}v_{i_j} \tag{6}$$

This is not the lowest variance estimator but has a better downstream performance (see Appendix E). We call it "oracle" because it assumes that the exact attention vector w is known, which is not true for sparse attention approximations.

Theorem B.2. *Oracle sampling estimation is unbiased and the trace of covariance is monotonically decreasing with $\mathcal{B}$.*

This theorem (proved in Appendix D) theoretically guarantees a low estimation error of oracle sampling. We also present an empirical comparison between oracle sampling estimation and TopK attention in Figures 7a and 7b. In summary, oracle sampling estimation can reduce relative error by up to $4\times$.

Note that the sampling budget $\mathcal{B}$ is not the actual computation cost for oracle sampling estimation: duplicate X_i need to be computed/loaded only once, so $\bar{o}$ can be computed by

$$\bar{o}=\sum_{i\in\mathcal{S}}\frac{f_i}{\mathcal{B}}v_i \quad S=\mathrm{Unique}(\{i_{1\leq i\leq\mathcal{B}}\}) \tag{7}$$

where f_i is the number of duplicates of X_i. Intuitively, if w has an peaked distribution (e.g. $w_i>99\%$), then almost all samples in $\{i_1,...,i_{\mathcal{B}}\}$ are identical to i. The actual computation cost of oracle sampling estimation is $|S|$, the number of *unique* samples, which we bound in the following:

Theorem B.3. *The expected computation budget ($\mathbb{E}(|S|)$) has an upper bound of $1+\mathcal{B}\epsilon$, where $\epsilon=1-\max_i w_i$.*

This theroem (proved in Appendix D) shows that the computation cost of oracle sampling is usually far less than the sampling budget. In Figure 7c, we present the downstream accuracy comparison between oracle sampling estimation and TopK attention. The former preserves high accuracy for both tasks, even with very small computation cost (0.002% out of 16k context, which is approximately 32).

C Algorithm intuition and explanation

C.1 Self-normalized importance sampling for attention estimation

Oracle sampling estimation cannot go beyond $2\times$ wall clock speed up because obtaining distribution w requires full computation of all qk_i^T and thereby only saving the wV computation.

Fortunately, importance sampling [Kloek and Van Dijk, 1978, Owen, 2013, Lohr, 2021] allows us to perform estimation for unknown distribution w by sampling from a proposed distribution u. In our problem setting, the normalization factor of w, i.e. $Z=\sum_{i=1}^{n}\exp\frac{qk_i^T}{\sqrt{d}}$ is also unknown because computing it requires evaluating all qk_i^T. However, we do have access to unnormalized weights $\widetilde{w_i}=e^{\frac{qk_i^T}{\sqrt{d}}}$ for sampled indices i. Hence, by employing a variant of importance sampling, **self-normalized importance sampling** [Owen, 2013], we sample indices $i_1,i_2,...,i_{\mathcal{B}}$ from a proposed distribution u and the resulting estimator is

$$X^{\mathrm{IS}}=\frac{1}{\widetilde{Z}}\sum_{j=1}^{\mathcal{B}}\frac{\widetilde{w_{i_j}}}{u_{i_j}}v_{i_j} \quad \text{where} \quad \widetilde{Z}=\sum_{j=1}^{\mathcal{B}}\frac{\widetilde{w_{i_j}}}{u_{i_j}} \tag{8}$$

which has a very nice property for accurately estimating attention output that $\mathbb{P}[\lim_{k\to\infty}X^{\mathrm{IS}}=o]=1$. Its variance[1] is related to the distribution u, and can be approximated by

$$\widetilde{\mathrm{Var}}(X^{\mathrm{IS}})=\frac{1}{\mathcal{B}}\mathbb{E}_{i\sim u}[\frac{w_i^2}{u_i^2}(v_i-o)^2]=\frac{1}{\mathcal{B}Z^2}\mathbb{E}_{i\sim u}[\frac{\widetilde{w_i}^2}{u_i^2}(v_i-o)^2] \tag{9}$$

To minimize the variance, u should satisfy $u\propto\widetilde{w_i}|v_i-o|$ [Hesterberg, 2003]. The variance will be high if u_i and $\widetilde{w_i}|v_i-o|$ assign a high probability mass to different regions of the sample space or have different modes. Therefore, the challenge is compute a distribution u aligned with $\widetilde{w_i}|v_i-o|$ without accessing too many $\widetilde{w_i}$. Besides, Equation (8) requires that sampling probability u can be computed and $u_i>0$, which is not satisfied by many deterministic approximations like TopK.

[1]We assume head dimension $d=1$ here for simplicity. Higher dimension has similar formulations and analysis by replacing variance with trace of covariance.

C.2 Variance reduction with LSH

We decompose $\widetilde{w_i}|v_i - o| = \exp(\frac{qk_i^T}{\sqrt{d}} + \log|v_i - o|)$. We observe emprically (Figure 9 in the appendix) that $\log|v_i - o|$ does not fluctuate significantly compared to $\frac{qk_i^T}{\sqrt{d}}$. Hence, we simplify the requirement of u to share the same peaks with qk_i^T. By the following transformation,

$$r = \max_{1 \le i \le n} |k_i| \quad \bar{q} = [q, 0] \quad \bar{k_i} = [k_i, \sqrt{r^2 - |k_i|^2}] \tag{10}$$

we further transfer inner product qk_i^T to cosine similarity between $\bar{q}$ and $\bar{k_i}$ (which is a common practice in Maximum Inner Product Search [Shrivastava and Li, 2014]).

Inspired by prior work [Spring and Shrivastava, 2017, Chen et al., 2020a], we leverage Locality sensitive hashing-based sampling for this estimation problem. Specifically, leveraging a hash function h in the LSH family that preserves cosine similarity such as SimHash [Sadowski, 2007], we can sample from probability distribution $u_i = \mathbb{P}[h(q) = h(k_i)]$ which is monotonic to $\cos\frac{qk_i^T}{|q| \cdot |k_i|}$.

D Proofs for theorems

D.1 Proof for Theorem B.2

Proof.

$$\mathbb{E}(\bar{o}) = \frac{1}{\mathcal{B}} \sum_{j=1}^{\mathcal{B}} \mathbb{E}[v_{i_j}] = \frac{1}{\mathcal{B}} \sum_{i=1}^{n} w_i v_i = o \tag{11}$$

Assume Σ_1 is the covariance matrix of $\bar{o}$, Σ_2 is the covariance matrix of v_i

$$\mathrm{Tr}(\Sigma_1) = \frac{1}{\mathcal{B}} \mathrm{Tr}(\Sigma_2) = \frac{1}{\mathcal{B}} (\mathbb{E}[||v_i||^2] - ||\mathbb{E}[v_i]||^2) = \frac{1}{\mathcal{B}} (\mathbb{E}[||v_i||^2] - ||o||^2) \tag{12}$$

$\mathbb{E}[||v_X||^2] - ||o||^2$ is a constant, so the trace of covariance matrix monotonically decreases with $\mathcal{B}$. □

D.2 Proof for Theorem B.3

Proof.

$$\mathbb{E}[|S|] = \mathbb{E}\Big[\sum_{i=1}^{n} \mathbf{1}_{i \in S}\Big] = \sum_{i=1}^{n} \mathbb{E}[\mathbf{1}_{i \in S}] = \sum_{i=1}^{n} (1 - (1 - w_i)^{\mathcal{B}}) = n - \sum_{i=1}^{n} (1 - w_i)^{\mathcal{B}} \tag{13}$$

Without loss of generality, let $a_i = 1 - w_i$ and $a_1 = \min_{1 \le i \le n} a_i = \epsilon$, then

$$\mathbb{E}[|S|] = n - \sum_{i=1}^{n} a_i^{\mathcal{B}} = n - a_1^{\mathcal{B}} - \sum_{i=2}^{n} a_i^{\mathcal{B}} \tag{14}$$

$$= n - \epsilon^{\mathcal{B}} - \sum_{i=2}^{n} a_i^{\mathcal{B}} \tag{15}$$

$f(x) = x^{\mathcal{B}}$ is convex function with $\mathcal{B} \ge 1$ and $x \ge 0$. Then with Jensen's inequality, we have

$$\sum_{i=2}^{n} a_i^{\mathcal{B}} \ge (n-1)\Big(\frac{\sum_{i=2}^{n} a_i}{n-1}\Big)^{\mathcal{B}} = (n-1)\Big(\frac{(\sum_{i=1}^{n} a_i) - a_1}{n-1}\Big)^{\mathcal{B}} \tag{16}$$

$$= (n-1)(\frac{n-1-\epsilon}{n-1})^{\mathcal{B}} = (n-1)(1 - \frac{\epsilon}{n-1})^{\mathcal{B}} \tag{17}$$

Let $g(x) = (1-x)^{\mathcal{B}} + \mathcal{B}x - 1$. We can prove $g(x) \ge 0$ for any $x \in (0,1), \mathcal{B} \ge 1$. Then we have

$$\sum_{i=2}^{n} a_i^{\mathcal{B}} \ge (n-1)(1 - \frac{\epsilon \mathcal{B}}{n-1}) = n - 1 - \epsilon \mathcal{B} \tag{18}$$

Then we finally have

$$\mathbb{E}[|S|] = n - \epsilon^{\mathcal{B}} - \sum_{i=2}^{n} a_i^{\mathcal{B}} \le 1 + \epsilon \mathcal{B} \tag{19}$$

□

Table 2: Comprehensive tasks on lm-eval-harness [Gao et al., 2021]. MAGICPIG significantly outperforms other methods with lower computation. The config (K,L) is hyper-parameter of LSH for MAGICPIG or page size and ratio of selected pages for Quest [Tang et al., 2024]. Cost_1, Cost_2 represents cost for searching/sampling and sparse attention computation respectively.

Methods	Config	GSM	COQA	MMLU	Avg.	Cost_1	Cost_2	Cost_{total}.
Llama-2-7b-chat	Full	22.4	75.8	49.2	49.1	0.00	1.00	1.00
MAGICPIG	(10,220)	17.3	76.4	48.6	**47.4**	0.00	0.04	0.04
MAGICPIG	(8,90)	18.7	75.0	47.9	47.2	0.00	0.08	0.08
Quest	(16,0.05)	13.0	69.4	41.4	41.3	0.06	0.05	0.11
Quest	(32,0.1)	15.7	70.2	44.0	43.3	0.03	0.10	0.13
Llama-3.1-8B-Instruct	Full	77.6	78.5	65.2	73.7	0.00	1.00	1.00
MAGICPIG	(10,220)	72.7	78.1	62.7	**71.2**	0.00	0.03	0.03
MAGICPIG	(8,90)	71.0	78.0	61.3	70.1	0.00	0.07	0.07
Quest	(16,0.05)	57.9	64.6	42.5	55.0	0.06	0.05	0.11
Quest	(32,0.1)	64.5	65.0	48.0	59.2	0.03	0.10	0.13

Table 3: Long context tasks on LongBench [Bai et al., 2023]. MAGICPIG preserves high accuracy with low computation. Config and cost are defined as in Table 2. Code models are only evaluated by Repobench-P and LCC.

Methods	Config	QaS	RbP	LCC	PrE	TrC	TrQ	Avg.	Cost_1	Cost_2	Cost_{total}.
Llama-3.1-8B-Instruct	Full	44.9	52.1	66.8	100.0	71.3	91.8	71.2	0.00	1.00	1.00
MAGICPIG	(10,150)	43.2	50.2	64.4	100.0	71.3	92.2	70.3	0.00	0.02	0.02
MAGICPIG	(8,75)	43.5	50.4	67.0	100.0	71.7	91.7	**70.7**	0.00	0.05	0.05
Quest	(16,0.05)	45.7	49.7	64.9	100.0	71.7	91.5	70.6	0.06	0.05	0.11
Quest	(32,0.1)	44.4	50.5	65.1	100.0	71.3	91.6	70.5	0.03	0.10	0.13
Code-Llama-13b-16K	Full		58.5	74.7				66.6	0.00	1.00	1.00
MAGICPIG	(10,150)		56.9	74.0				**65.5**	0.00	0.03	0.03
Quest	(16,0.05)		56.4	74.4				65.4	0.06	0.10	0.11

E Oracle sampling

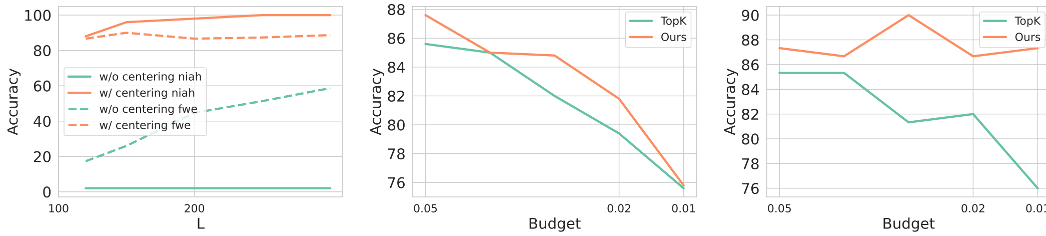
The optimal sampling probability to guarantee estimation is unbiased in terms of lowest variance is not directly using attention score distribution w_i, but $u_i' \propto w_i||v_i||$. However, this sampling probability is not optimal in terms of downstream accuracy and efficiency. We attribute this to two reasons. First, we observe the value norm of the sink token is significantly smaller than others (Figure 10), given its lower probability of being sampled, which may influence the functionality of attention. Second, due to the same reason, $u_i' \propto w_i||v_i||$ is flatter than w_i, resulting larger computation cost (as analyzed by Theorem B.3).

F Additional Evaluation

Additional evaluation includes 3 mid-context comprehensive tasks from lm-eval-harness [Gao et al., 2021] (GSM8K-CoT [Cobbe et al., 2021], MMLU-Flan-Cot-Fewshot [Hendrycks et al., 2020] and COQA [Reddy et al., 2019]), and 6 long context tasks from [Bai et al., 2023] (QASPER [Dasigi et al., 2021], LCC, Repobench-P [Liu et al., 2023], TriviaQA [Joshi et al., 2017], PRE and TREC [Li and Roth, 2002, Hovy et al., 2001]). Results are in Tables 2 and 3. Compared with Quest, which also shows reasonable performance on long context tasks, MAGICPIG also demonstrates good performance on tasks with moderate context sizes in lm-eval-harness [Gao et al., 2021], indicating a more robust performance in general serving.

G Hardware Configuration

Our CPU is Intel(R) Xeon(R) Platinum 8480+ for A100 and Intel(R) Xeon(R) Gold 6338 CPU @ 2.00GHz for L40. In the last setting, the CPU bandwidth is estimated at 100GB/s which is above the empirical bandwidth we measure when running a group query attention of size 4. We simulate 24GB GPU by setting memory limit with L40. As the bandwidth of L40 (864GB/s) is less than RTX 4090 (1TB/s), the real speed of our system should be slightly faster than simulation.



(a) Importance of centering **(b)** CWE **(c)** FWE

Figure 8: Left: Accuracy comparison for with and without centering. Here we fix K and vary L for the two settings. **Mid and Right:** Comparison between TopK attention and MAGICPIG. In the two aggregated tasks, sampling based MAGICPIG can even beat the exact TopK attention. The experiments are done on RULER [Hsieh et al., 2024] with 16K context size.

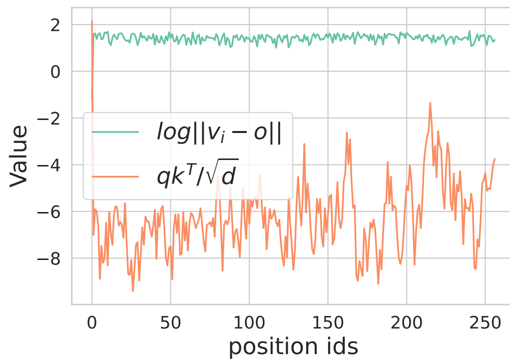


Figure 9: The range of fluctuation of $\log|v_i - o|$ and $\frac{qk_i^T}{\sqrt{d}}$ in a single decoding step. Compared to $\frac{qk_i^T}{\sqrt{d}}$, $\log|v_i - o|$ is stable, hence we do not consider $\log|v_i - o|$ in our proposed sampling probability.

H Ablation Study

In this section, we empirically validate our two previous observations.

Centering is important for good performance. In Section 2.1, we use a translation to center the keys before applying LSH sampling. Empirical results show this to be important for downstream tasks as shown in Figure 8a. Without centering, the accuracy drops to almost zero in retrieval (NIAH) and degrades to 65% in FWE. We find almost none keys (less than 0.1%) can be sampled by query without centering, as their orientation is almost opposite as shown in Figure 2c.

Sampling goes beyond TopK**.** In Figures 8b and 8c, We compare the performance of MAGICPIG and TopK attention in two aggregated tasks (CWE, FWE) where TopK attention experiences significant performance degradation (Figure 1). MAGICPIG can even beat exact TopK attention in these two tasks by a margin up to 3% and 8% respectively, demonstrating that sampling improves the ceiling of TopK, which is impossible for a search-only algorithm.

I Supplementary analysis

Figure 9 shows that compared to $\frac{qk_i^T}{\sqrt{d}}$, $\log|v_i - o|$ is stable in a decoding step.

Figure 10 shows that the norm of the value states of attention sink is smaller than others.

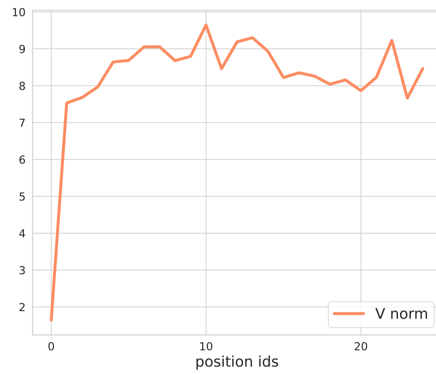


Figure 10: The y-axis is the norm of values states $\|v_i\|$ for token i (on the x-axis). We observe that the value norm $\|v_0\|$ of the attention sink is significantly smaller than others.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

Justification:

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [No]

   Justification:

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.

- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification:

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.

- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [NA]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).

- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification:

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.