RetrievalAttention

Accelerating Long-Context LLM Inference via Vector Retrieval

Paper by Di Liu, Meng Chen, Baotong Lu, et al.

arXiv:2409.10516v3 Paper discussion, 28 October 2026

Published at NeurIPS 2025 Main Conference

One question: how can decoding read less KV?

The setting

Long prompt already prefetched; decode one new token at a time with limited GPU memory.

- Full KV cache grows with context length; each decode step must access history.
- Attention weights are sparse, but the important positions change with the query.
- RetrievalAttention stores most KV on CPU and searches a query-aware index for each step.

Code status (28 Oct 2026): the Microsoft repository currently hosts RetroInfer; an implementation of the original method was not verified.

Paper, Sections 1–3, PDF pp. 1–7. All quantitative results here refer to arXiv v3.

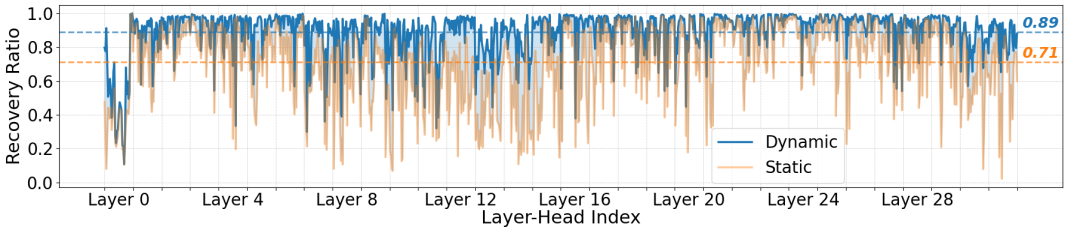
The bottleneck is both storage and KV access

Prompt length	128K	256K	512K	1M
Full KV capacity (GB)	15.6	31.2	62.5	125
No-cache token latency (s)	32.8	111	465	1,765
Attention share (s)	25.2	96	434	1,695

- GPU-resident KV becomes infeasible on a 24 GB RTX 4090.
- Recomputing the prompt avoids the cache, but is prohibitively slow.

Llama-3-8B; paper, Table 1, PDF p. 2. Capacity and no-cache latency describe different configurations.

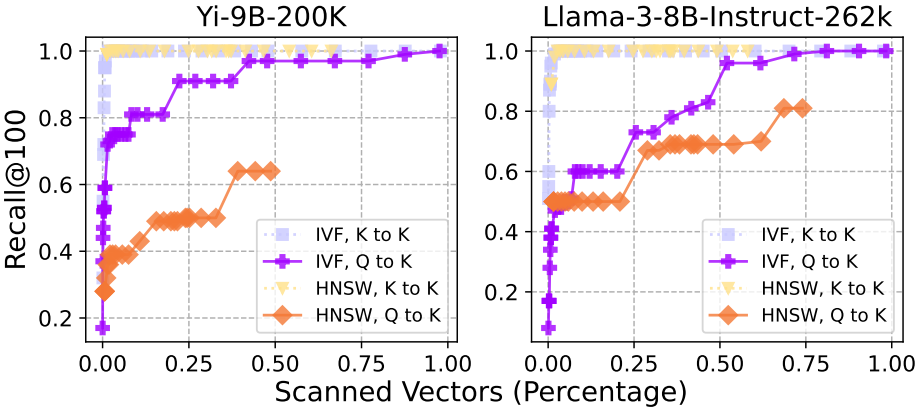
The active tokens are sparse *and* dynamic



- Dynamic top-1000 recovers 89% of attention mass on average.
- Freezing the first token's top-1000 drops recovery to 71%.

100K-token KV retrieval, 20 generated tokens, Llama-3-8B; paper, Fig. 2, PDF p. 4.

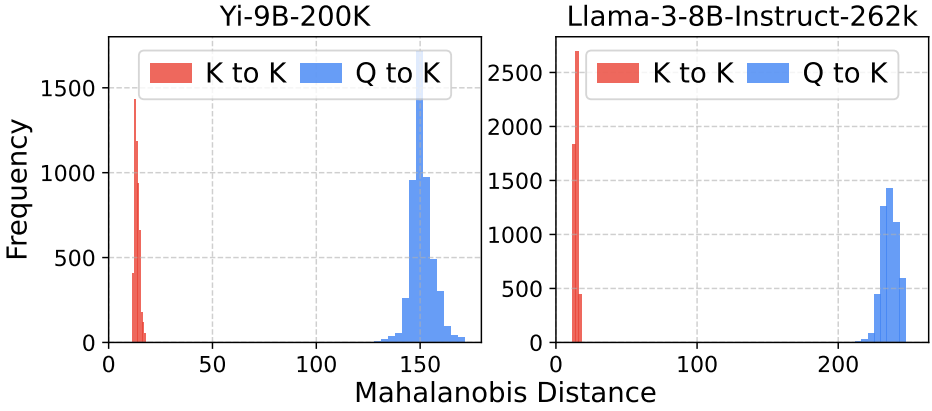
Ordinary ANN indices miss Q-to-K neighbors



- Key-to-key search is easy; actual query-to-key search is out of distribution.
- IVF scans roughly 30–50% of keys for Recall@100 above 0.95.

Paper, Fig. 3a and Section 2.4, PDF pp. 4–5.

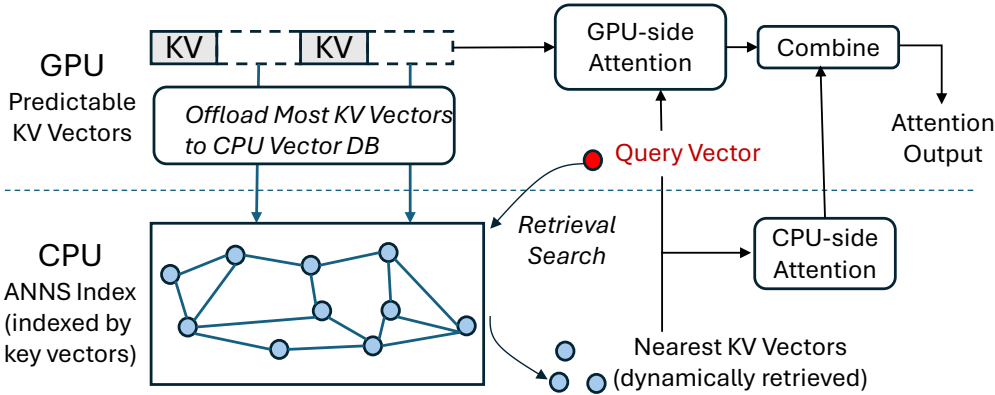
Query and key vectors occupy different regions



- The Mahalanobis-distance probe puts Q-to-K far beyond K-to-K.
- A graph built only from K-to-K proximity follows the wrong geometry.

Yi-9B and Llama-3-8B, 128K-token prompt; paper, Fig. 3b, PDF pp. 4–5.

RetrievalAttention splits the work across CPU and GPU



- GPU: a small predictable KV set.
- CPU: indexed historical KV, searched with the current query.
- Compute partial attention in parallel; merge with correct normalization.

Paper, Fig. 4a and Section 3, PDF pp. 5–7.

Sparse attention renormalizes selected tokens

$$z_i = \frac{\mathbf{q}_t \mathbf{k}_i^\top}{\sqrt{d}}, \qquad a_{t,i} = \frac{e^{z_i}}{\sum_{j \le t} e^{z_j}}$$

$$\mathcal{I}_{t,\epsilon} = \{ i : a_{t,i} > \epsilon \}, \qquad \mathbf{o}_t \approx \sum_{i \in \mathcal{I}_{t,\epsilon}} \frac{e^{z_i}}{\sum_{j \in \mathcal{I}_{t,\epsilon}} e^{z_j}} \mathbf{v}_i$$

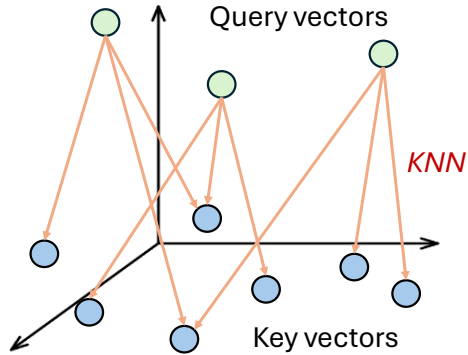
- Search aims to find the high-weight KV pairs.
- The implementation uses a retrieval budget (usually top-100), not an oracle threshold.

Paper, Eqs. (1)–(2), PDF pp. 3 and 6; retrieval budgets in Table 2, PDF p. 8.

Prefill queries teach the key graph what matters

1. Use prefill queries to find exact top-k keys on GPU.
2. Connect each query to those keys.
3. Project shared-query links into a key-only graph.
4. Search this graph with each new decode query.

The projection follows the RoarGraph idea and avoids retaining query nodes.



Paper, Section 3.2 and Fig. 4b, PDF pp. 5–6. Figure shows the pre-projection Q–K links.

The GPU keeps 640 fixed positions

Memory	KV set	Decode work
GPU	First 128 + last 512 tokens	FlashAttention on $\mathcal{W}$
CPU	Most other historical tokens	Index search + attention on Ω

- $\mathcal{W}$ and Ω are disjoint; top-100 is the default dynamic budget.
- Only retrieved CPU KV participates in the CPU partial attention.
- Multiple query heads search in parallel; GQA heads get separate indices but share KV storage.

Paper, Section 3.3, PDF pp. 6–7; Table 2, PDF p. 8; Appendix C, PDF p. 18.

Two local softmax outputs need global weights

$$Z_S = \sum_{i \in S} e^{z_i - m_S}, \quad \mathbf{o}_S = \frac{\sum_{i \in S} e^{z_i - m_S} \mathbf{v}_i}{Z_S}, \quad m_S = \max_{i \in S} z_i$$

$$\gamma_S = \frac{e^{m_S - m} Z_S}{e^{m_{\mathcal{W}} - m} Z_{\mathcal{W}} + e^{m_\Omega - m} Z_\Omega}, \quad \mathbf{o} = \gamma_{\mathcal{W}} \mathbf{o}_{\mathcal{W}} + \gamma_\Omega \mathbf{o}_\Omega$$

What this guarantees

The merge equals softmax over the *selected union* $\mathcal{W} \cup \Omega$. Any approximation to full attention comes from omitted KV.

Derived stable form of paper, Eqs. (3)–(5), Appendix B, PDF p. 17; $m = \max(m_{\mathcal{W}}, m_\Omega)$.

The evaluation separates task quality from latency

- **RTX 4090 testbed:** 24 GB GPU, 10-core i9-10900X, 128 GB DRAM; batch size 1.
- **Models:** Llama-3-8B-Instruct-262k, Yi-6B-200K, Yi-9B-200K.
- **Quality:** seven ∞-Bench tasks, RULER at 4K–128K, needle retrieval.
- **Search:** Recall@100 against exact top-100 and fraction of keys scanned.
- **Speed:** seconds per generated token, after long-context prefill.

Paper, Section 4.1, PDF p. 7; Tables 2–5, PDF pp. 8–10.

Baselines test different ways to reduce KV cost

Family	Representative methods and role
Full attention	No-cache recomputation; vLLM with GPU KV cache
KV reduction	StreamingLLM, SnapKV, InfLLM, Quest, InfiniGen
Vector search	Flat: exact linear scan; IVF: key-cluster ANN

- Flat and IVF isolate the benefit of the query-aware index.
- Faster fixed-budget cache methods must be compared together with accuracy.

Paper, Section 4.1, PDF p. 7.

InfiniteBench averages stay near full attention

Model	Full	Retrieval top-100	Retrieval top-2000
Llama-3-8B	50.4	48.9	49.6
Yi-9B	52.8	50.8	52.2
Yi-6B	45.5	45.0	45.0

- Seven-task average; higher is better.
- Top-2000 results are reported for the hard KV retrieval task; other task settings remain as in Table 2.

Paper, Table 2, PDF p. 8. Scores are percentage points, not relative speedups.

The hard KV task exposes the retrieval budget

∞-Bench Retr.KV	Full	Top-100	Top-2000
Llama-3-8B	17.5	9.0	14.0
Yi-9B	30.5	20.0	30.0
Yi-6B	3.5	3.0	3.0

- More retrieved KV can recover accuracy on difficult associative lookup.
- Near-full average accuracy does not mean every task matches full attention.

Paper, Table 2, PDF p. 8. Higher is better.

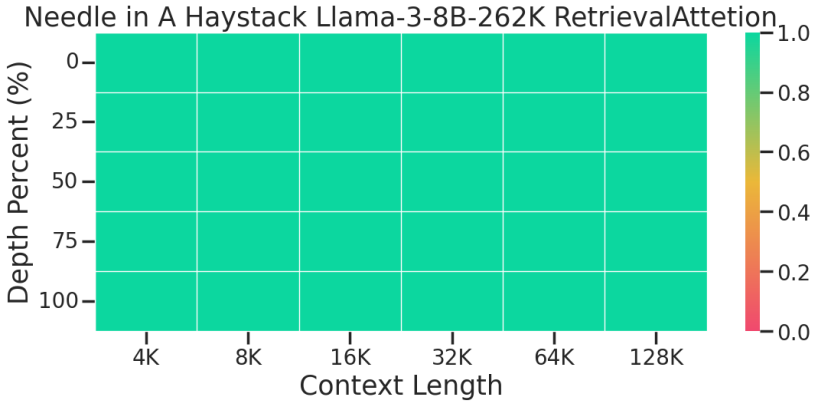
RULER is close on average, with 128K gaps

Model	Average, 4K–128K		128K only	
	Full	Retrieval	Full	Retrieval
Llama-3-8B	86.54	84.70	78.74	74.70
Yi-9B	76.87	76.43	60.51	59.15
Yi-6B	67.86	65.86	55.77	51.44

- Scores are task percentages; higher is better.
- Table 3 also reports a shorter effective context than full attention for Llama-3-8B: 16K vs 32K.

Paper, Table 3, PDF p. 9.

Needle retrieval works across tested positions



- All displayed 4K–128K depth/length cells pass on Llama-3-8B.
- Appendix A.3 also reports passing 250K–1M cells with Llama-3-8B-1048K.

Paper, Figs. 5 and 8, PDF pp. 9 and 16. Synthetic needle retrieval.

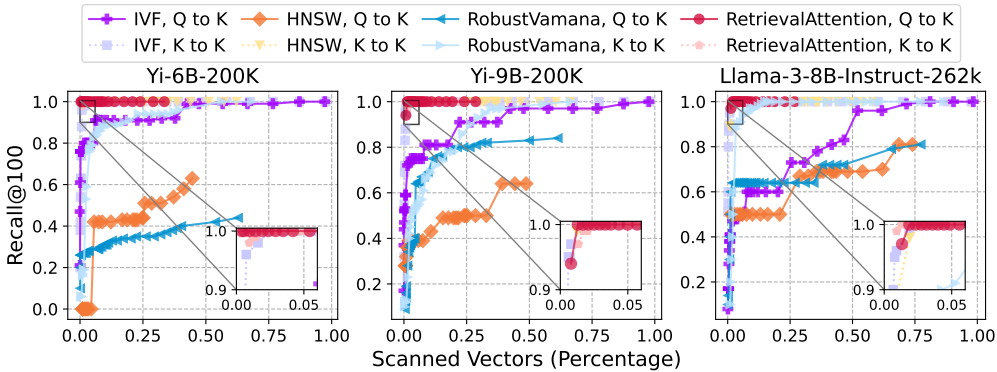
At 128K, index search drives the speedup

Method	Retrieval	Attention	Other	Total (s/token)
Flat	0.798	0.083	0.041	0.922
IVF	0.250	0.084	0.039	0.373
RetrievalAttention	**0.064**	0.081	0.043	**0.188**

- Total latency: 4.9× lower than Flat; 1.98× lower than IVF.
- The attention component barely changes; retrieval is the differentiator.

Llama-3-8B, RTX 4090, 128K, batch size 1; paper, Tables 4–5, PDF pp. 9–10.

Low scan cost retains high top-100 recall



- RetrievalAttention reaches Recall@100 > 0.95 after scanning about 1–3% of keys.
- The advantage appears on Q-to-K; K-to-K is easy for several indices.

Paper, Fig. 6 and Section 4.4, PDF pp. 9–10. Scanned fraction differs from top-100 output size.

Available GPU memory changes the comparison

A100, 128K (s/token)	Llama-3-8B	Yi-6B	Yi-9B
vLLM, full GPU KV	0.033	0.030	0.044
IVF	0.345	0.309	0.468
RetrievalAttention	0.155	0.150	0.227

- With enough GPU memory, vLLM's full cache is faster.
- On the paper's RTX 4090 testbed, vLLM is reported OOM even at 4K.

Paper, Appendix A.4, Table 7, PDF p. 16; RTX 4090 Table 4, PDF p. 9. Different machines.

Million-token latency uses a large-memory server

Llama-3-8B (s/token)	100K	200K	500K	1M
Flat	0.489	0.871	1.92	3.69
IVF	0.308	0.476	1.032	1.889
RetrievalAttention	0.159	0.167	0.170	0.172

- The 1M result uses an A100 80 GB and a machine with **1.72 TB DRAM**.
- It is a separate hardware regime from the 24 GB RTX 4090 result.

Paper, Appendix A.4, Table 8, PDF p. 16.

Extra experiments show where budget matters

Adaptive allocation at 128K

PyramidKV-style layer budget:

- Retr.KV: 14.5 $\rightarrow$ 16.0
- Seven-task mean: 49.9 $\rightarrow$ 50.1
- En.QA: 8.7 $\rightarrow$ 8.5

70B stress test

Eight A100 40 GB GPUs, KV retrieval:

- Accuracy: 23.5 vs 24.0 Flat
- Full attention accuracy: 35.0
- Latency: 1.62 vs 5.68 s/token Flat

Paper, Appendix F–G, Tables 10–11, PDF p. 19. Adaptive-budget run has its own setup.

The result is a specific accuracy–resource tradeoff

1. Query-aware graph construction addresses the measured Q/K distribution gap.
2. CPU retrieval reduces scanned KV; GPU retains the cheap predictable positions.
3. The merge is exact on selected KV, while selection quality controls task accuracy.

Deployment conditions reflected in the paper

Prefill/index building is outside the primary decode latency; large contexts need CPU DRAM; hard retrieval tasks may need a larger top-k budget.

Paper, Sections 3–4 and Appendices A, B, G, PDF pp. 5–10, 16–19.

Paper identity and links

- **Read version:** arXiv:2409.10516v3, revised 31 Dec 2024.
- **Publication:** NeurIPS 2025 Main Conference paper.
- **Code check (28 Oct 2026):** original RetrievalAttention implementation not verified. The Microsoft repository currently implements the later RetroInfer system.
- **Version caution:** proceedings abstract says 0.107 s/token; this presentation uses the arXiv v3 value of 0.188 s/token.

Paper, arXiv v3 PDF p. 1 and Table 4, PDF p. 9; linked proceedings and repository.

Backup: the normalization identity

$$N_S = \sum_{i \in S} e^{z_i - m_S} \mathbf{v}_i, \qquad Z_S = \sum_{i \in S} e^{z_i - m_S}, \qquad \mathbf{o}_S = N_S / Z_S.$$

$$\frac{\sum_{S \in \{\mathcal{W}, \Omega\}} e^{m_S - m} N_S}{\sum_{S \in \{\mathcal{W}, \Omega\}} e^{m_S - m} Z_S} = \sum_{S \in \{\mathcal{W}, \Omega\}} \underbrace{\frac{e^{m_S - m} Z_S}{\sum_T e^{m_T - m} Z_T}}_{\gamma_S} \mathbf{o}_S.$$

- $m = \max(m_{\mathcal{W}}, m_{\Omega})$ prevents overflow.
- Disjoint $\mathcal{W}, \Omega$ give the exact softmax on their union.

Algebraic expansion of paper, Appendix B, Eqs. (3)–(5), PDF p. 17.