

Conference Talk (10 min)

Flux Attention: Context-Aware Hybrid Attention for Efficient LLMs Inference

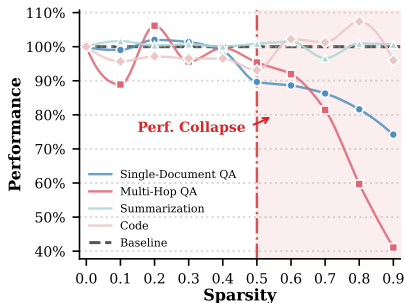
Qiu et al. — NeurIPS 2026, arXiv:2604.07394v2

Quantong Qiu¹ Zhiyi Hong¹ Yi Yang¹ Haitian Wang¹
Kebin Liu² Qingqing Dang² Juntao Li^{1,†} Min Zhang¹

¹Soochow University ²Baidu Inc.

Static hybrid attention breaks on retrieval-heavy tasks

One fixed FA/SA ratio cannot serve tasks whose retrieval demands differ.



Qiu et al., Fig. 1(a): accuracy vs. model sparsity Ω_{MSR} (SSA sweep, LongBench tasks).

Observation

- Retrieval-intensive tasks (QA) **collapse** once sparsity crosses a task-specific threshold.
- Context-holistic tasks (Summarization, Code) tolerate high sparsity.

Consequence: static hybrids [5, 1] pick one ratio for all inputs — **degrade retrieval** or **waste compute**.

Fine-grained sparsity is hardware-unfriendly at decode

Head-level sparsity barely accelerates decoding; layer-level sparsity does.

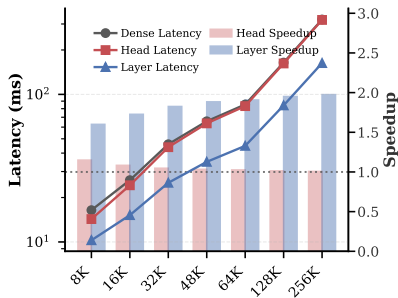


Fig. 1(b): decode latency (log) and speedup vs. context length.

Why head-level fails

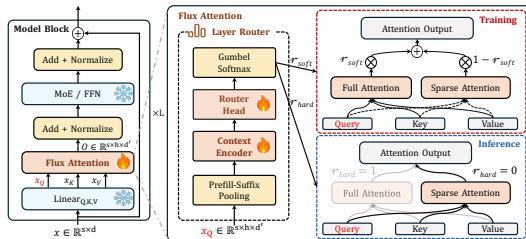
- Mixed workloads $\Rightarrow$ intra-layer **imbalance**.
- Fast blocks idle on a **sync long-tail**; FLOP cuts $\nrightarrow$ wall-clock.

Why layer-level works

- Uniform work, contiguous access.
- Sparse layers **skip historical KV loading entirely**.

Flux Attention: route whole layers, not heads

A lightweight Layer Router sends each transformer layer to FA or SA by context.



Qiu et al., Fig. 2: one Flux Attention block (router + dual attention paths), repeated $\times L$; backbone stays frozen.

- Left to right: $x_Q \rightarrow$ **Layer Router** $\rightarrow r_{\text{soft}}$ mixes FA / SA (training); r_{hard} picks one path (inference).
- Only the router trains — the backbone stays **strictly frozen**.

Layer Router: soft to train, hard to serve

Gumbel-Softmax makes discrete routing differentiable; inference is argmax.

Router pipeline

- **Prefill-Suffix Pooling** on the boundary 100 tokens $\rightarrow$ one sequence descriptor.
- **Context Encoder** (MLP) $\rightarrow$ **Router Head** $\rightarrow$ logits $\pi_{\text{FA}}, \pi_{\text{SA}}$.

Training (soft, Gumbel-Softmax [3])

$$r_{\text{soft}} = \frac{e^{(\pi_{\text{FA}} + g_{\text{FA}})/\tau}}{e^{(\pi_{\text{FA}} + g_{\text{FA}})/\tau} + e^{(\pi_{\text{SA}} + g_{\text{SA}})/\tau}}$$

$$\mathcal{O} = r_{\text{soft}} \text{FA} + (1 - r_{\text{soft}}) \text{SA}$$

Temperature τ anneals high $\rightarrow$ low.

Inference (hard)

- $r_{\text{hard}} = \arg \max$: each layer runs **either FA or SA**.
- Routing runs **once at prefill**; the cached decision is reused for every decode step $\Rightarrow$ no per-token router cost.

Why not route every token?

Overhead + broken per-layer homogeneity; task intent is stable per response.

Training: language loss + sparsity budget, router only

A Lagrangian budget keeps sparsity near target without hard-coding it.

$$\max_{\lambda_1, \lambda_2 \geq 0} \min_{\theta} \underbrace{L_{\text{language}}}_{\text{quality}} + \underbrace{\lambda_1 L_{\text{diff}} + \lambda_2 L_{\text{diff}}^2}_{\text{sparsity regularization}}, \quad L_{\text{diff}} = \mathbb{E}[1 - r_{\text{soft}}] - t$$

- θ : **router parameters only**; backbone frozen; λ_1, λ_2 per-task multipliers via gradient ascent.
- Budget t is **task-dependent and non-tight** (bounds): the realized Ω_{MSR} is reported, not the target.
- Deployment: sparse layers keep **minimal KV**; retrieval layers keep full KV.

Cost

12 h on 8×A800 per run; 5 data sources, ~0.74B tokens, 1K–64K.

Quality holds across long-context and reasoning benchmarks

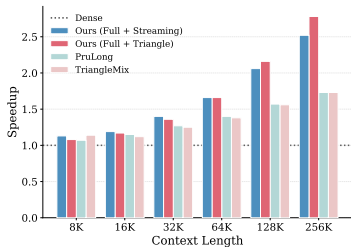
FluxAttn matches full attention and beats sparsity baselines on average.

Backbone	Method	LongBench-E avg.	RULER avg. (256K)
Qwen3-4B	Full attention	48.45	66.00 (43.27)
	+ DuoAttn / PruLong / TriMix	48.22 / 47.16 / 47.72	60.67 / 60.25 / 63.74
	+ FluxAttn (FA-TA)	48.76	66.95 (56.91)
	+ FluxAttn (FA-SSA, sp. decode)	48.59	67.19 (56.00)
Llama-3.1-8B	Full attention	53.28	83.47 (72.34)
	+ DuoAttn / PruLong / TriMix	52.11 / 51.68 / 51.67	70.33 / 59.87 / 80.46
	+ FluxAttn (FA-XA)	53.07	79.51 (60.80)

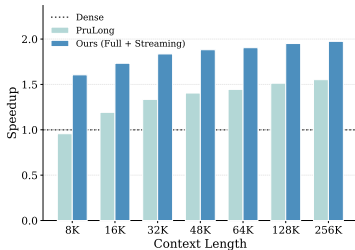
- Qwen3-4B sparse decode: **best RULER avg. in group** (67.19 vs. 66.00 dense).
- Llama-3.1-8B stays **close to backbone** (53.07 vs. 53.28); RULER below backbone — sparsity is not free.
- Also best-in-group on LongBench-v2, GSM8K, AIME24.

Speedups grow with context length

Up to $2.8\times$ prefill and $\approx 2.0\times$ decode at 256K context.



Qiu et al., Fig. 3(a): prefill speedup (TTFT) vs. dense.

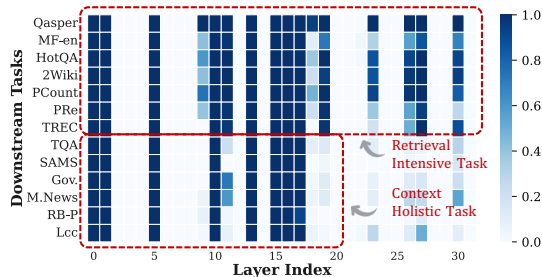


Qiu et al., Fig. 3(b): per-token decode speedup vs. dense.

- 256K, FA-TA: **2.8** $\times$ prefill speedup, ahead of PruLong and TriangleMix.
- Decode approaches **2.0** $\times$ at 256K — the Frame-3 layer-level benefit, realized end to end.
- Panels measure **different phases and units**; the decode claim is per-token.

What the router actually learns

Routing is task-aware and input-aware, and keeps universal layers dense.



Qiu et al., Fig. 4: FA activation frequency per layer, retrieval-intensive (top) vs. context-holistic tasks.

- **Task-level:** retrieval tasks keep many FA layers (dark); holistic tasks route mid/high layers to SA.
- **Input-level:** mid-range frequencies (~ 0.4 – 0.6) show per-instance adaptation, not task memorization.
- **Universal core:** layers 0, 1, 5, 13, 15–17 stay FA across all tasks — backbone structure is preserved.

Takeaways

Context-aware layer routing turns sparsity into real wall-clock gains.

1. **Granularity matters:** route whole layers, not heads — hardware-friendly, contiguous memory access at decode.
2. **Cheap to add:** frozen backbone, one router; 12 h on 8×A800.
3. **Evidence:** LongBench-E / RULER averages at or above full attention on Qwen3; up to $2.8\times$ prefill and $\approx 2.0\times$ decode at 256K.

Boundaries (as reported by the authors)

RULER on Llama-3.1-8B remains below its backbone (79.51 vs. 83.47); decode speedups are kernel-level; matched-sparsity and serving results in backup.

Thank you

Flux Attention: route layers by context, keep retrieval dense.

Thank you

Questions and discussion

Paper: [arXiv:2604.07394v2](https://arxiv.org/abs/2604.07394v2) Code: github.com/qqtang-code/FluxAttention2026

Backup: matched sparsity $\Omega_{\text{MSR}} \approx 0.50$ (Llama-3.1-8B)

At equal sparsity, Flux beats PruLong on accuracy, latency, and memory.

Context	Accuracy		Decode latency	Persistent KV
	Flux – PruLong	vs. dense FA2	vs. PruLong	memory vs. PruLong
128K	+14.76 pts	comparable	–19.2%	–16.1% to –39.3%
256K	+29.39 pts	comparable	–32.0%	–16.1% to –39.3%

Authors' caveat: a “sparse unit” is a head for PruLong vs. a layer for Flux, so realized Ω_{MSR} is reported with measured latency and memory; the comparison reflects the optimized head-wise implementation evaluated.

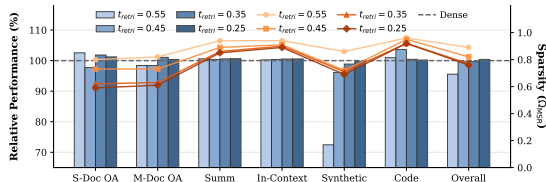
Backup: serving behavior and router overhead

Gains survive batching; the router itself is length-invariant.

- **Batched serving** (32K context, $8\times$ NVIDIA H20): Flux is faster at every feasible batch size — **$1.32\times$ at batch 8, $2.07\times$ at batch 16** — and reaches batch 24 where dense FA2 exhausts memory.
- **Router overhead**: averages **0.20 ms per layer**, constant from 512 to 1M tokens — routing does not become the bottleneck at extreme lengths.

Backup: sparsity target and continued training

The budget is a soft dial; the backbone can adapt to fixed routes.



Qiu et al., Fig. 5: performance (bars) and realized Ω_{MSR} (line) per task under different t_{retri} .

- Lower t_{retri} = more FA on retrieval tasks; 0.45 adopted as the balance.
- Set too high: sharp drops on retrieval tasks, matching Frame 2.
- **Continued training** with the router frozen: Qwen3-4B/8B pass their backbones within 50 steps; Llama recovers steadily.

References

-  Adithya Bhaskar, Alexander Wettig, Tianyu Gao, Yihe Dong, and Danqi Chen.
Cache me if you can: How many kvs do you need for effective long-context lms?
arXiv preprint arXiv:2506.17121, 2025.
-  Zhiyuan He, Yike Zhang, Chengruidong Zhang, Huiqiang Jiang, Yuqing Yang, and Lili Qiu.
Trianglemix: Accelerating prefilling via decoding-time contribution sparsity, 2025.
-  Eric Jang, Shixiang Gu, and Ben Poole.
Categorical reparameterization with gumbel-softmax.
arXiv preprint arXiv:1611.01144, 2016.
-  Quantong Qiu, Zhiyi Hong, Yi Yang, Haitian Wang, Kebin Liu, Qingqing Dang, Juntao Li, and Min Zhang.
Flux attention: Context-aware hybrid attention for efficient llms inference, 2026.
v2; NeurIPS 2026.
-  Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han.
Duoattention: Efficient long-context llm inference with retrieval and streaming heads.
arXiv preprint arXiv:2410.10819, 2024.