LeapQuant

Efficient Linear Attention with Accurate Recurrent State Quantization

Paper by Yi Pan, Haocheng Xi, et al.

Independent research group discussion

arXiv:2609.38166v1 | 29 September 2026

Reading prepared 19 October 2026

Quantize less often; protect the dominant state structure.

A fixed-size state still costs bandwidth and capacity



- Every decoded token loads and updates a dense state.
- Prefix caching stores many state checkpoints: one per 1,024 cached tokens in this experiment.

Paper, Fig. 1, PDF p. 2. Left: GLM-5.3-Flash-NVFP4, B200, batch 256. Right: Qwen3.5-9B memory with prefix caching.

One recurrence describes both GDN and KDA

$$S_t = \mathrm{Diag}(\alpha_t) S_{t-1} + k_t \big(v_t - S_{t-1}^\top \beta_t\big)^\top, \qquad o_t = S_t^\top q_t$$

State and reads

$S_t \in \mathbb{R}^{d_k \times d_v}$

$q_t, k_t, \beta_t \in \mathbb{R}^{d_k}$

$v_t, o_t \in \mathbb{R}^{d_v}$

A compact update record

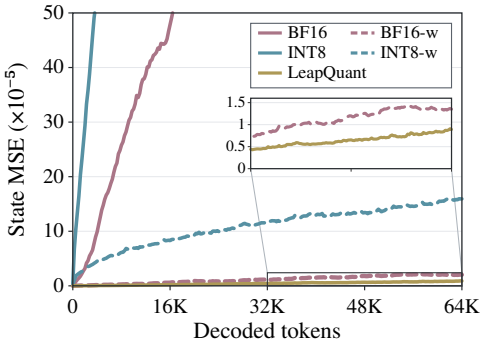
$u_t = v_t - S_{t-1}^\top \beta_t$

Store (α_t, k_t, u_t): diagonal decay plus a rank-one write.

β_t here is a read vector; native GDN/KDA notation also uses a scalar gate called β_t.

Paper, Eqs. (1), (3), PDF p. 4; architecture mappings in Appendix A, PDF p. 13.

Rounding errors become part of the next state



At fixed layer inputs,

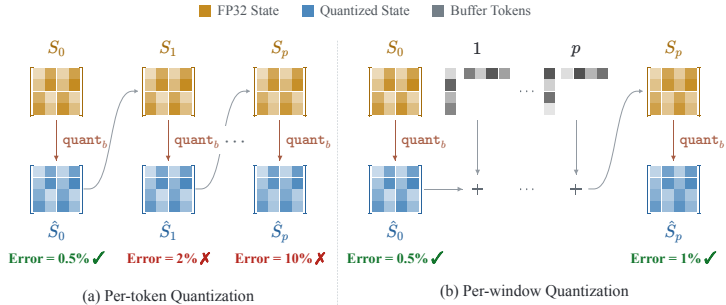
$$E_t = A_t E_{t-1} + \varepsilon_t,$$

$$A_t = \mathrm{Diag}(\alpha_t) - k_t \beta_t^\top.$$

- Per-step rounding injects ε_t at every token.
- At 64K, windowing lowers MSE by 100× for BF16 and 39× for INT8.

Paper, Fig. 3 / Sec. 3.2, PDF p. 4; Eq. (8), PDF p. 14. Qwen3.5-9B on PG-19; -w means a 16-token window. A_t is a presentation shorthand.

Hold the boundary state fixed for 16 tokens



$$S_\ell = \Gamma_{1:\ell}\hat{S}_0 + \sum_{j=1}^{\ell} \Gamma_{j+1:\ell} k_j u_j^\top$$

$\Gamma_{a:b}$ is the product of diagonal decays; quantize the full state only at $\ell = p$.

Paper, Fig. 2, PDF p. 3; Eq. (4), PDF p. 4. Error percentages in this illustration are schematic, not measured results. Dequantization of $\hat{S}_0$ is implicit.

Read the state without materializing it every step

$$S_\ell^\top x = \hat{S}_0^\top \Gamma_{1:\ell} x + \sum_{j=1}^{\ell} u_j \left(k_j^\top \Gamma_{j+1:\ell} x\right)$$

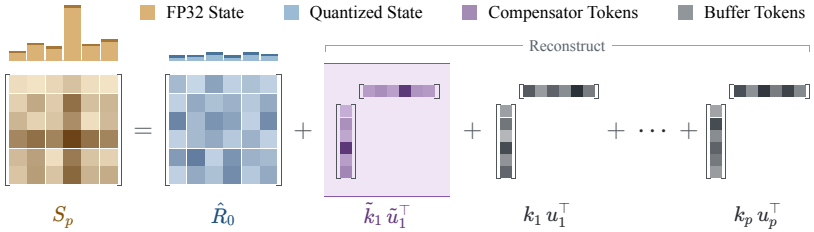
1. Read $S_{i-1}^\top \beta_i$ to compute the correction u_i.
2. Append (α_i, k_i, u_i) to the buffer.
3. Read $S_i^\top q_i$ to produce the current output.

Cost and latency

Base read: $O(d_k d_v)$; each record: $O(d_k + d_v)$.
Output is produced every token. The base state is still read every token.

Paper, Appendix B.2, Eq. (9), PDF p. 14; window start set to zero here. Cumulative decays are maintained incrementally.

Compensator Tokens preserve a few dominant directions

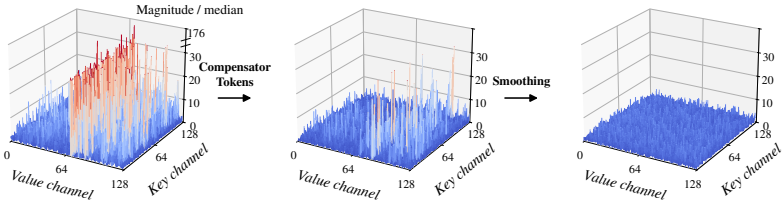


$$R_p = S_p - \tilde{K}\tilde{U}^\top, \qquad \tilde{S}_p = \mathrm{dequant}_b(\hat{R}_p) + \tilde{K}\tilde{U}^\top$$

- Each vector pair is a rank-one record using the real-token path.
- Default: $r = 4$ FP16 pairs, refitted at every boundary by power iteration.
- These records have no text content and produce no output.

Paper, Fig. 4 / Eqs. (5)–(6), PDF p. 5; implementation, PDF p. 7. The schematic omits the decays shown in the full reconstruction equation.

Smooth the residual before assigning quantization levels



$$c_a \simeq \sqrt{\frac{1}{d_v}\sum_j |(R_0)_{aj}|}, \quad C = \mathrm{Diag}(c), \quad \hat{R}_0^C = \mathrm{quant}_b(C^{-1}R_0)$$

- Use a positive scale floor; recompute C at each boundary.
- Restore C on the residual read. Other records keep their original coordinates.

Paper, Fig. 6 / Sec. 3.4, PDF p. 6. Panels (a),(b) normalize by the original-state median; (c) uses its own median. The displayed scale formula omits the positive floor.

The complete readout is the sum of three projections

$$S_\ell^\top x = \underbrace{\mathrm{dequant}_b(\hat{R}_0^C)^\top C \Gamma_{1:\ell} x}_{\text{quantized residual}} + \underbrace{\tilde{U}(\tilde{K}^\top \Gamma_{1:\ell} x)}_{\text{high-precision compensators}} + \underbrace{\sum_{j=1}^{\ell} u_j (k_j^\top \Gamma_{j+1:\ell} x)}_{\text{buffered real-token updates}} .$$

All contributions are combined in the original state coordinates.

Paper, Appendix B.3, Eq. (10), PDF p. 14. $\tilde{K} \in \mathbb{R}^{d_k \times r}$ and $\tilde{U} \in \mathbb{R}^{d_v \times r}$.

Boundary work is amortized across the window

1. Reconstruct the current state from residual, compensators, and records.
2. Fit new low-rank factors; replace the old compensators.
3. Smooth and quantize the new residual.
4. Clear the real-token buffer and start the next window.

Default implementation

$p = 16$, $r = 4$; FP16 compensators and update buffer, FP32 smoothing scales.
At 4 bits, use $r = 8$. vLLM integration with TileLang decode kernels.

No retraining or calibration data; boundary computation remains a runtime cost.

Paper, Secs. 3.3–3.4, PDF pp. 5–6; Sec. 4.1, PDF p. 7.

Quality uses three complete models and shared prompts

Model	Recurrence	Evaluation
Qwen3.5-9B	GDN	Quality + efficiency
Qwen3.5-35B-A3B	GDN	Quality + efficiency
Kimi-Linear-48B-A3B	KDA	Quality + efficiency
Qwen3.8-Flash	GDN	8 layers, efficiency only
GLM-5.3-Flash	KDA	8 layers, efficiency only

- Main tasks: AIME 2026, GPQA-Diamond, LCB v6, MMLU-Pro.
- Three-seed averages; identical prompts and sampling within each model.
- Quantization baselines re-quantize every token; other components keep baseline precision.

Paper, Sec. 4.1, PDF pp. 6–7; sampling details in Table 6, PDF p. 16. Kimi's full name ends with Instruct. LCB = LiveCodeBench.

Eight-bit LeapQuant matches the FP32 average

12 model–task pairs

State method	Avg. (%)
FP32	75.5
BF16, per-step	68.7
LeapQuant 8-bit	75.5
FP8, per-step	34.2
INT8, per-step	26.3
KVQuant, adapted	67.0
QuaRot, adapted	50.1
TurboQuant, adapted	49.6

Qwen3.5-9B on AIME

FP32	87.9
BF16	72.1
Per-step INT8	7.1
LeapQuant	87.9

Individual 8-bit scores differ from FP32 by at most 0.8 percentage points across the 12 pairs.

Paper, Table 1, PDF p. 7. Higher is better. Average: 3 models × 4 tasks; excludes GSM8K. The 0.8-point range is calculated from the reported entries.

Four bits remain useful, but Qwen loses substantial quality

State bits	LeapQuant avg.	Strong comparator	Comparator avg.
8	75.5	KVQuant	67.0
6	72.4	TurboQuant	31.9
4	60.4	TurboQuant	22.9

- FP32 average: 75.5%. Four-bit LeapQuant uses $r = 8$.
- Kimi at 4 bits: every main task stays within 1.1 percentage points of FP32.
- Qwen3.5-35B-A3B at 4 bits: AIME 91.5 $\rightarrow$ 59.2; LCB 75.6 $\rightarrow$ 37.4.

Interpretation

Near-lossless quality is the strongest at 8 bits; lower precision is model- and task-dependent.

Paper, Table 1, PDF p. 7; Sec. 4.2, PDF p. 8. Scores are percentages.

Large batches expose the benefit of lower state traffic

Single-layer kernel speedup at batch 512

GPU	GDN	KDA
NVIDIA B200	2.68×	2.41×
RTX PRO 6000	3.95×	3.38×
RTX 5090	4.25×	3.64×

- One layer, 32 heads, $d_k = d_v = 128$; baseline is the FLA FP32-state kernel.
- Larger batches make recurrent-state traffic a stronger bottleneck.
- These numbers measure the attention kernel, not the complete model.

Paper, Fig. 7 / Sec. 4.3, PDF p. 8. Values transcribed at the specified batch size; the abstract reports averaged ranges.

Whole-model decode gains are smaller than kernel gains

Model	B200, batch 512	PRO 6000, max batch
Qwen3.5-9B	1.23×	1.27× (256)
Qwen3.5-35B-A3B	1.24×	1.41× (256)*
Kimi-Linear-48B-A3B	1.22×	1.22× (256)
GLM-5.3-Flash, 8 layers	1.37×	1.42× (512)
Qwen3.8-Flash, 8 layers	1.33×	1.57× (512)

- vLLM with CUDA graphs; context is 4K except *2K.
- Small batches benefit less: Qwen35B on B200 at batch 32 is 1.00×.
- Other attention layers, projections, and MoE computation remain.

Paper, Fig. 8 / Sec. 4.3, PDF p. 8. Baseline uses FP32 recurrent states. The two Flash models are truncated efficiency models.

The boundary state takes 1.19 bytes per element

Representation accounting: 128×128 state, $r = 4$

INT8 residual + FP16 factors + row/value scales:

$$1 + \frac{2r(d_k + d_v)}{d_k d_v} + \frac{4(d_k + d_v)}{d_k d_v} = 1.1875 \text{ bytes/element.}$$

Approximately $3.4\times$ smaller than FP32, before active-request buffers.

- With a checkpoint per 1K cached tokens, reported total memory falls by 41%, 51%, and 56% for Qwen9B, Qwen35B, and Kimi.
- Qwen9B on one B200 supports up to $1.4\times$ as many concurrent requests.

Paper, Sec. 4.3, PDF pp. 8–9; scale layout in Appendix B.3, PDF p. 14. Formula is a size calculation with FP32 vector scales, not a separate memory measurement.

Offline output throughput improves by about 1.47×

GPU	Model	GSM8K	MMLU-Pro	GPQA
B200	Qwen9B	1.23	1.30	1.44
	Qwen35B	1.43	1.60	1.58
	Kimi48B	1.54	1.60	1.47
PRO 6000	Qwen9B	1.40	1.38	1.31
	Qwen35B	1.65	1.59	1.41
	Kimi48B	1.48	1.51	1.49

- Entries are output-throughput ratios to FP32-state vLLM.
- Faster decode and more concurrent requests both contribute.
- Kimi uses Qwen9B's input/output length distribution.

Paper, Fig. 9 / Sec. 4.3, PDF p. 9. Average of the 18 rounded ratios is approximately 1.47; this is not a latency metric.

Windowing repairs most of the damage; all parts matter

Qwen3.5-9B state method	AIME	LCB	Kernel speedup
FP32 per-step	87.9	64.1	1.00×
BF16 per-step	72.1	49.6	1.64×
BF16 + window	87.8	62.9	1.71×
INT8 per-step	7.1	9.2	2.43×
+ Window	82.4	60.6	2.64×
+ Compensator Tokens	86.6	61.5	2.56×
+ Smoothing	87.9	64.1	2.52×

Added compensation and smoothing trade a little kernel speed for recovered quality.

Paper, Table 2, PDF p. 9. Accuracy in percent; kernel ratios measured at batch 256 against FP32.

Bigger windows and more compensators are not always better

Vary p, fix $r = 4$

p	AIME	LCB	Speedup
4	86.2	62.8	1.59×
8	87.1	63.2	2.13×
16	87.9	64.1	2.52×
32	87.9	64.2	2.20×

Larger buffers increase traffic and shared-memory use.

Vary r, fix $p = 16$

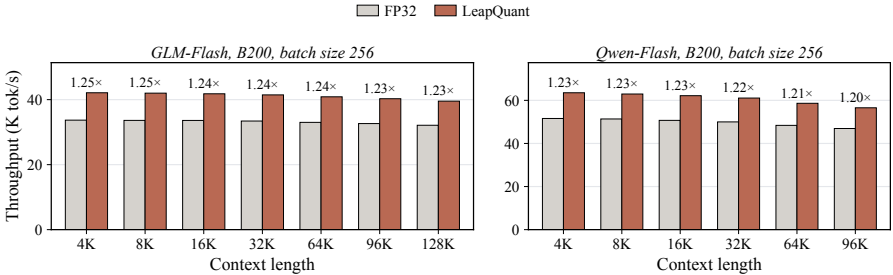
r	AIME	LCB	Speedup
2	85.7	63.4	2.57×
4	87.9	64.1	2.52×
8	88.1	64.0	2.37×
16	87.9	62.8	0.74×

Fitting and reconstruction costs become exposed as rank grows.

$p = 16$, $r = 4$ balances accuracy and speed in this ablation.

Paper, Table 3, PDF p. 9. Qwen3.5-9B; accuracy in percent; batch 256 kernel ratios.

Sparse hybrid attention retains gains at long context



- B200, batch 256; both models use 8-layer efficiency versions.
- GLM: 1.23× at 128K. Qwen: 1.20× at 96K.
- With dense full attention, its growing cost dilutes the gain; Kimi still reaches 1.18× at 8K, batch 512.

Paper, Appendix D, Figs. 10–11, PDF p. 16. These are throughput measurements, not long-context accuracy results.

The useful regime follows the state cost and precision budget

Strongest evidence

Near-FP32 quality at 8 bits on complete GDN/KDA models; throughput benefits at larger batches and with prefix checkpoints.

- Four-bit quality depends strongly on the model and task.
- Small batches and expensive unmodified layers reduce end-to-end gains.
- The general recurrence covers more architectures than the evaluated models.
- Fixed-input error propagation explains a mechanism; it is not a full-model error bound.

Evidence: Tables 1–3, PDF pp. 7–9; Fig. 8, PDF p. 8; Appendices A–B, PDF pp. 13–14. Interpretation synthesized for this discussion.

Quantization timing is part of the memory representation

1. **Time:** keep recent updates in higher precision and quantize the full state only at window boundaries.
2. **Structure:** preserve dominant components as rank-one records.
3. **Coordinates:** smooth the residual and restore its scale during reads.

Takeaway

The main contribution is a practical combination of accurate recurrent storage and an efficient read/update path.

Yi Pan et al., LeapQuant, arXiv:2609.38166v1

Backup slides follow: mappings, detailed scores, scales, and memory accounting.

Paper, Sec. 5, PDF p. 10; synthesis based on the method and ablations.

Backup: GDN and KDA fit the same update form

Appendix notation:

$$S_t = D_t S_{t-1} + \tilde{k}_t (v_t - S_{t-1}^\top b_t)^\top.$$

Model	D_t	$\tilde{k}_t$	b_t
DeltaNet	I	$\beta_t k_t$	k_t
GDN	$\alpha_t I$	$\beta_t k_t$	$\alpha_t k_t$
KDA	$\mathrm{Diag}(\alpha_t)$	$\beta_t k_t$	$\alpha_t \odot k_t$

- In this table, k_t, β_t are the native key and scalar update gate.
- RWKV7 uses two records: an erase sub-step and a separate write.
- DeltaProduct uses multiple delta-rule sub-steps per token.

Paper, Appendix A / Table 4, PDF p. 13. The main-text write/read vectors are $\tilde{k}_t, b_t$ in the appendix.

Backup: eight-bit accuracy across all 12 main pairs

Model	State	AIME	GPQA	LCB	MMLU-Pro
Qwen9B	FP32	87.9	81.3	64.1	83.3
	LeapQuant	87.9	81.8	64.1	83.8
Qwen35B	FP32	91.5	84.7	75.6	85.9
	LeapQuant	91.0	83.9	76.1	85.8
Kimi48B	FP32	67.5	70.3	41.4	72.4
	LeapQuant	68.3	69.8	41.3	72.1

GSM8K (FP32 $\rightarrow$ LeapQuant):

Qwen9B 96.1 $\rightarrow$ 96.3; Qwen35B 96.7 $\rightarrow$ 96.5; Kimi 92.1 $\rightarrow$ 92.0.

Paper, Table 1, PDF p. 7; Table 5, PDF p. 15. Scores in percent, higher is better; means over three seeds.

Backup: the four-bit penalty differs sharply by model

Model	State	AIME	GPQA	LCB	MMLU-Pro
Qwen9B	FP32	87.9	81.3	64.1	83.3
	LeapQuant 4-bit	58.1	65.7	34.0	80.2
Qwen35B	FP32	91.5	84.7	75.6	85.9
	LeapQuant 4-bit	59.2	57.6	37.4	81.4
Kimi48B	FP32	67.5	70.3	41.4	72.4
	LeapQuant 4-bit	67.9	69.2	40.5	73.4

Four-bit configuration: $p = 16$, $r = 8$. Qwen thinking traces are longer; the paper links this to greater sensitivity to recurrent error.

Paper, Table 1 / Sec. 4.2, PDF pp. 7–8. Scores in percent; the trace-length explanation is the authors' interpretation.

Backup: keep key and value scales on the residual path

Per-value-channel symmetric INT8:

$$\mathrm{dequant}_8(\hat{R}_0^C) = \frac{Z_0}{127} B, \qquad C = \mathrm{Diag}(c).$$

Then

$$\left(C \frac{Z_0}{127} B \right)^\top \Gamma x = B \frac{Z_0^\top}{127} C \Gamma x.$$

- Z_0: integer payload; B: value-side scale matrix.
- C and cumulative decay Γ act on key coordinates.
- Compensator and real-token records are already in original coordinates.

Paper, Appendix B.3, PDF p. 14. Multiplying residual scales into other records would change the represented recurrence.

Backup: generation settings are shared within each model

Setting	Both Qwen3.5 models	Kimi-Linear Instruct
Temperature	1.0	1.0
Top-p	0.95	1.0
Top-k	20	–
Presence penalty	1.5	0
Thinking	on	–
Maximum output tokens	81,920	65,536

- Same prompts, sample set, and sampling settings for all methods of a model.
- Maximum model length = output limit + 8,192 tokens.

Paper, Appendix E / Table 6, PDF p. 16. Kimi has no thinking mode; this table describes quality evaluation.

Backup: active-request buffers add to checkpoint storage

Record type	Values per token	FP16 bytes at $p = 16$
GDN	$1 + d_k + d_v$	8,224
KDA, explicit vector decay	$2d_k + d_v$	12,288

Here $d_k = d_v = 128$; the buffer is allocated per active request, head, and layer.

- The main text quotes approximately 8 KiB, consistent with GDN's record count.
- The KDA figure above is a calculation from the appendix's explicit record layout.
- A cached boundary checkpoint does not need a full active-window buffer.

Paper, memory discussion, PDF p. 8; Table 4, PDF p. 13. Buffer sizes calculated from the stated record counts; not measured allocator usage.

Reference and figure attribution

LeapQuant: Efficient Linear Attention with Accurate Recurrent State Quantization

Yi Pan, Haocheng Xi, Kan Zhu, Xingyang Li, Yibo Wu,
Mayank Mishra, Hongtao Zhang, William X. Zheng,
Baris Kasikci, Song Han, Kurt Keutzer, Rishabh Iyer, Ion Stoica.

arXiv:2609.38166v1, submitted 29 September 2026

Original figures reproduced from the authors' source package.
Paper license: Creative Commons Attribution 4.0.

This deck is an independent reading presentation.
Quantitative tables are transcribed or explicitly calculated from this version.