SparseGDN: Chunk Sparse in Linear Attention

Jiedong Zhuang, Peng Zhao, Jialin Tang, Haoji Hu
Zhejiang University and ByteDance

Central idea

Approximate local Delta solves, select persistent memory writes, and reconstruct outputs for every token from compact states.

arXiv:2610.41420v1, submitted 30 October 2026

Publication not found as of 2026-11-03 (arXiv record).
Public code not found as of 2026-11-03.

Research group discussion | 3 November 2026

Linear-time GDN prefill still performs expensive dense work

Stage	Dense work at every chunk and value head
Intra	Local Delta solve; transformed values U and correction W
H	Read correction $U - WS$; recurrent state update and snapshots
O	Historical-state reads and causal local interactions

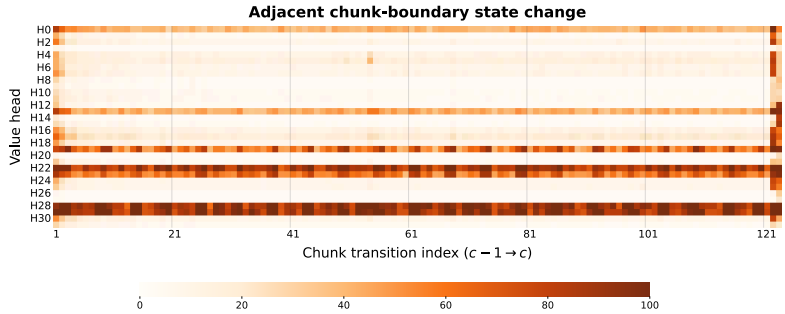
Question

Can fewer recurrent events and less state traffic accelerate pretrained GDNs while retaining long-context task quality?

Reported speedups cover the GDN core, including online selection.

Paper, Secs. 1–2.2, PDF pp. 2–4; timing scope, Sec. 3.1, p. 12.

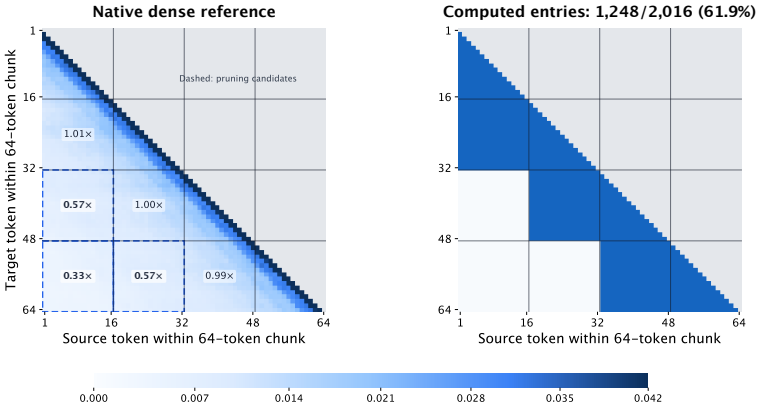
State changes are small for many heads and large for a few



- Median relative change: 6.56%; 63.8% of transitions are below 10%.
- 21 of 32 heads have median change below 10%.
- Motivates different recurrent budgets across head pairs.

Paper, Fig. 2a, PDF p. 2. One 8K RULER input, Qwen3.5-4B layer 16. Color: relative state change.

Local inverse coefficients concentrate near the causal diagonal



Keep diagonal and adjacent 16-token blocks inside each 64-token chunk.

Paper, Fig. 2b, PDF p. 2. Left: mean $|M^{-1} - I|$ over 123 chunks and 32 heads; right: retained support.

Dense GDN corrects values before writing residual information

$$U'_c = U_c - W_c S_c$$

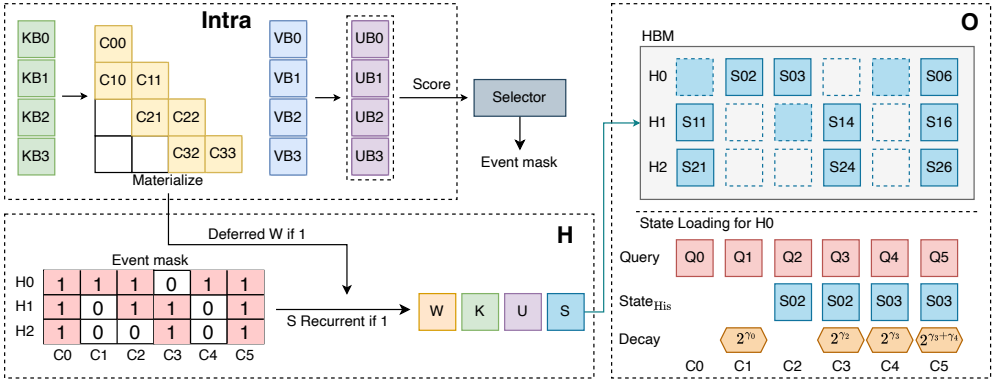
$$S_{c+1} = 2^{\gamma_c} S_c + K_c^\top \left(U'_c \odot 2^{\gamma_c - g_c}\right)$$

$$O_c = \frac{1}{\sqrt{d}} \left[2^{g_c} \odot (Q_c S_c) + A_c U'_c\right]$$

- $S_c \in \mathbb{R}^{128\times 128}$: state entering chunk c.
- g_c: cumulative log-base-two retention; $\gamma_c = g_{c,63}$.
- A_c: causal query–key interactions with relative decay.

Paper, Sec. 2.1, PDF p. 4, Eqs. (1)–(4). Head index omitted; $\odot$ scales token rows.

Five stages turn an event mask into work actually avoided



- Intra computes U, compact coefficients, and scores for all chunks.
- Selected events alone materialize W and execute recurrent read/write.

Paper, Fig. 3, PDF p. 3; approximation versus execution boundary, Sec. 2.2, p. 4.

Seven inverse blocks replace a full 64-token Delta solve

$$MU = \mathrm{diag}(\beta)V, \qquad MW = \mathrm{diag}(\beta \odot 2^{g})K$$

$$M^{-1} \approx C = \begin{pmatrix} C_{00} & 0 & 0 & 0 \\ C_{10} & C_{11} & 0 & 0 \\ 0 & C_{21} & C_{22} & 0 \\ 0 & 0 & C_{32} & C_{33} \end{pmatrix}$$

$$C_{ii} = M_{ii}^{-1}, \qquad C_{i,i-1} = -C_{ii} M_{i,i-1} C_{i-1,i-1}$$

- Retained blocks are exact inverse blocks in exact arithmetic.
- 1,248 / 2,016 = 61.9% of causal off-diagonal entries remain.

Paper, Sec. 2.3, PDF p. 5, Eqs. (5)–(8). Each block is 16×16; omitted distant blocks cause approximation.

Tail-16 ranks chunks before recurrent propagation

$$s_{c,h} = \left(\sum_{t=48}^{63} \sum_{v=0}^{d_v-1} U_{c,h,t,v}^2 \right)^{1/2}$$

$$q_c = \frac{1}{2} \sum_{h \,\text{in pair}} \frac{s_{c,h}}{\|s_{\cdot,h}\|_2}$$

- Late tokens undergo less attenuation before the chunk boundary.
- Tail-16 fits the final value tile already resident on chip.
- Per-head normalization balances the two heads for chunk ranking.

Ranking proxy

The true write also depends on keys, decay, and the correction WS.

Paper, Secs. 2.3–2.4.1, PDF pp. 5–7, Eqs. (9)–(10).

Budget upgrades favor persistent and visible memory

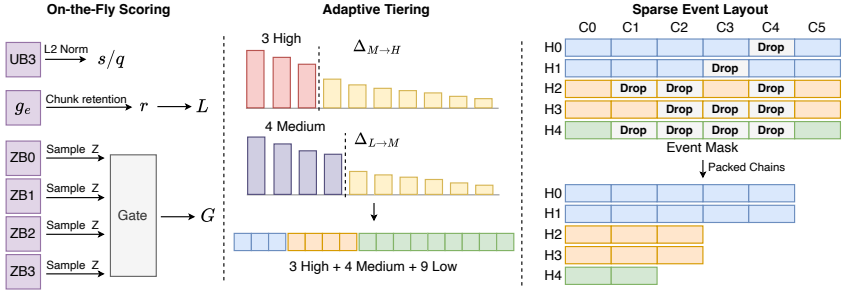
$$L_h = \sum_{j=1}^{C-2} r_h^j, \qquad G_h = \sqrt{\eta_h / \bar{\eta}}$$

$$\Delta_{a \to b} = \sum_{c \in \mathcal{C}_b \setminus \mathcal{C}_a} \sum_{h \text{ in pair}} s_{c,h} L_h G_h$$

- r_h: mean end retention over the last at most 16 chunks.
- L_h: geometric persistence proxy over a $C - 2$ chunk horizon.
- η_h: sampled energy of SiLU output gates and RMSNorm scales.
- Upgrade gains use raw tail magnitudes across head pairs.

Paper, Sec. 2.4.2, PDF pp. 7–8, Eqs. (11)–(12). $C = \lceil T/64 \rceil$. Later Delta updates are outside the lifetime proxy.

The selector adapts both pair budgets and chunk identities



$$(\rho_L, \rho_M, \rho_H) = (0.08, 0.30, 0.95)$$

- First and last chunks are anchors; other chunks follow q_c.
- Assign high tiers, then medium tiers, using upgrade-gain separation.
- Nominal mean keep: 28.44–31.31%; measured mean: about 29%.

Paper, Fig. 5 and Sec. 2.4, PDF pp. 7–8. Shown 9-low/4-medium/3-high counts are illustrative.

Deferred materialization removes work before the event scan

$$W_i = \overline{C}_{ii} K_i + \overline{C}_{i,i-1} K_{i-1}, \qquad \overline{C}_{ij} = C_{ij}\,\mathrm{diag}(\beta_j \odot 2^{g_j})$$

	Dense	SparseGDN
Materialize W	Every chunk/head	Selected events only
Recurrent layout	Dense chunk sequence	CSR chain per head pair
Intermediate states	Every boundary	Referenced event snapshots

- CSR rows prioritize high-, medium-, then low-retention chains.
- Preallocated capacity $E_{\max} = 16k_H$ avoids host synchronization.

Paper, Sec. 2.5, PDF pp. 8–9, Eqs. (15)–(16). The first block has no preceding-block term.

Skipping an event preserves decay while omitting read/write

A skipped chunk

$$U'_{c,h} = U_{c,h}, \qquad S_{c+1,h} = 2^{\gamma_{c,h}} S_{c,h}$$

$$P_{0,h} = 0, \qquad P_{c+1,h} = \sum_{j=0}^{c} \gamma_{j,h}$$

$$S_{c,h} = 2^{P_{c,h} - P_{e+1,h}} S_{e+1,h} \quad \text{for consecutive retained events } e < c$$

- Scan retained events and bridge each gap with one accumulated decay.
- Overwrite U with U' only at retained events.
- Return the final state separately for autoregressive decoding.

Paper, Sec. 2.6, PDF pp. 9–10, Eqs. (17)–(21). Zero-initialized prefill.

A skipped even chunk still contributes to its neighbor's output

Group $(c, c+1) = (2r, 2r+1)$ reuses the entry state S_c:

$$O_{c+1} = \frac{1}{\sqrt{d}}\big[2^{g_{c+1}+\gamma_c} \odot (Q_{c+1} S_c) + A_{c+1,c} U'_c + A_{c+1,c+1} U'_{c+1}\big].$$

$$O_{c+1} - O^{\text{chunk}}_{c+1} = \frac{1 - z_c}{\sqrt{d}} A_{c+1,c} U'_c$$

- If $z_c = 0$, this local contribution bypasses persistent state.
- It also stays outside the next chunk's recurrent read correction.
- Group-local reads are part of the sparse operator definition.

Paper, Sec. 2.7, PDF pp. 10–11, Eqs. (22)–(25). Comparison fixes Intra results and the event mask.

Three evaluation scopes answer different questions

Quality	13 RULER tasks, six lengths 4K–128K; 100 examples/task/length; 7,800 examples/model
Core latency	Intra+H+O versus Selector+Intra+H+O; fixed workload at each length; 24-layer total
Peak memory	Complete-model BF16 prefill, batch 1, cache enabled; last-token logits only; up to 256K

- Qwen3.5-4B/9B: 24 GDN layers, 16 Q/K heads, 32 value heads.
- BF16 activations and matrix inputs; FP32 state accumulation/statistics.
- Core timings exclude projections, attention layers, MLPs, and decoding.

Paper, Sec. 3.1, PDF p. 12; Sec. 3.7, pp. 19–20. Exact GPU model is unspecified in final text.

Default retention keeps aggregate RULER quality near baseline

Model	Native	Sparse	Δ (pp)	Keep (%)
4B	96.142	96.159	+0.017	29.044
9B	96.617	96.502	−0.115	29.043

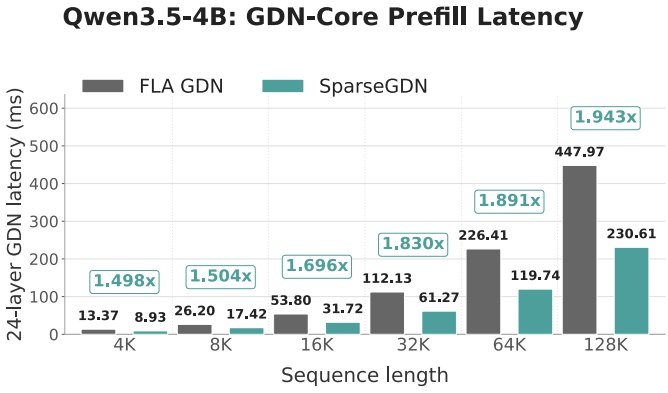
- 4B: largest absolute length-level change is +0.341 pp at 16K.
- 9B: largest drop is −0.692 pp at 8K.
- For the other five 9B lengths, $|\Delta| \leq 0.103$ pp.

Units

Scores are percentages; arithmetic score differences are percentage points (pp), computed before rounding.

Paper, Table 1, PDF p. 12; Sec. 3.2, p. 13. Aggregate averages the six lengths.

Core prefill speedup grows with context length



- 4B, 128K: 447.97 → 230.61 ms; 1.943× speedup.
- 9B, 128K: 453.34 → 235.02 ms; 1.929× speedup.

Paper, Fig. 1, PDF p. 1; Table 1, p. 12. Batch 1; all 24 GDN layers; online selection included.

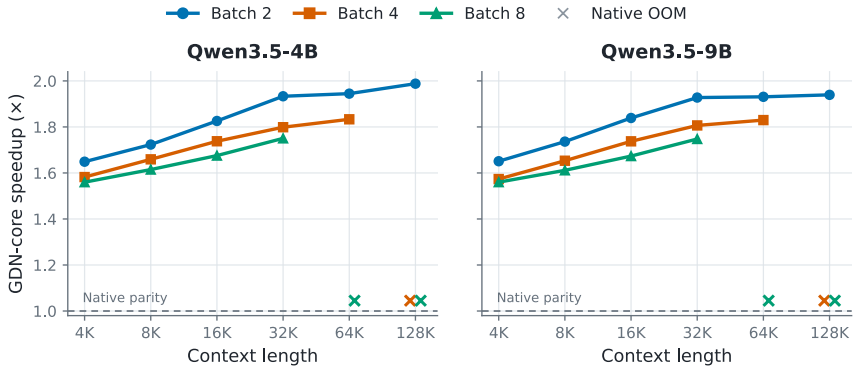
Each stage benefits, while every token still needs an output

4B at 128K	Native (ms)	Sparse (ms)	Ratio
Intra / Intra+Selector	157.04	63.23	2.484×
H	161.18	80.61	1.999×
O	129.74	86.66	1.497×

- W construction moves from native Intra to selected-event H.
- H saves corrections and writes; O saves state traffic.
- Local interactions and token outputs remain dense in O.

Paper, Fig. 7, PDF p. 13. Per-stage medians; their sum need not equal a median total.

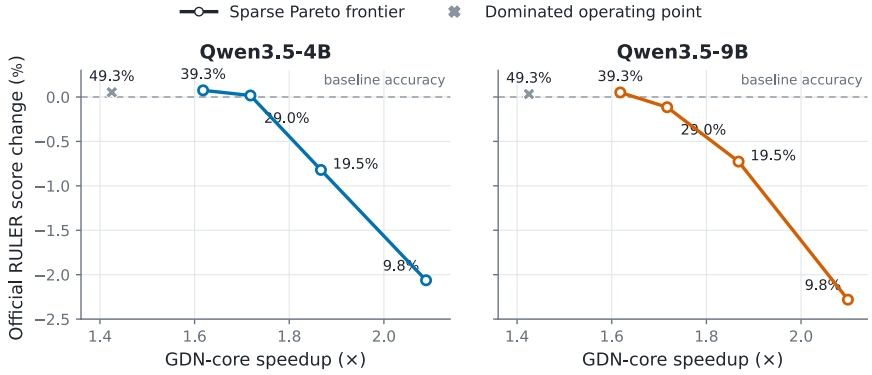
Batched execution remains faster at every completed paired point



- Batch 2/4/8: 1.560×–1.988× GDN-core speedup.
- Crosses indicate Native OOM; their plotted height is arbitrary.

Paper, Fig. 8 and Table 2, PDF p. 14. Sparse completion is independently verified for only two OOM points.

The retention budget sets a measurable speed–quality trade-off



- About 9.8% keep: $\sim 2.1\times$ speedup, ~ 2 pp score loss.
- About 39.3% keep: $1.618\times$, with small positive observed deltas.

Paper, Table 3, PDF p. 15; Fig. 9, p. 16. Speedup: geomean over six lengths. Vertical differences: pp.

Matched-budget random selection loses useful information

4K RULER score (%)	4B	9B
Native	96.641	96.821
Block-adjacent + all	96.410	96.846
Full64 + adaptive	96.669	96.787
Block-adjacent + random	95.067	95.621
SparseGDN	96.377	96.846

- Adaptive beats random by 1.310 / 1.226 pp at about 29% keep.
- Across six lengths: advantages of 2.006 / 1.614 pp.

Paper, Table 4, PDF p. 17; Sec. 3.5, pp. 15–16. Custom variants share group-local reads; adaptive masks are reselected.

Dynamic allocation improves the accuracy–retention trade-off

9B, all six lengths	Score (%)	Keep (%)
Fixed tier counts 9/4/3	96.183	29.816
Dynamic tier counts	96.502	29.043

Paired score gain

$+0.319$ pp; 95% bootstrap CI $[0.114, 0.532]$ pp.

- Pair identities remain input-dependent in the fixed-count baseline.
- Realized keep rates are reported separately and are not matched.
- Tail-16 balances ranking quality with the final on-chip tile.

Paper, Table 5, PDF p. 17; Tail-window analysis, pp. 17–18. 20,000 stratified paired resamples.

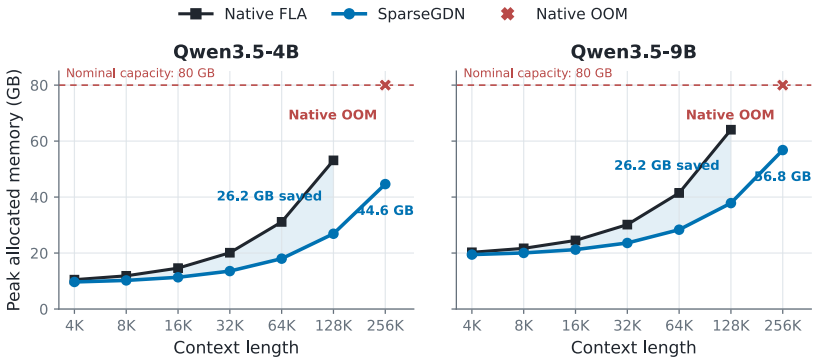
Less materialization and state traffic contribute to speed

Optimization	Scope	4B	9B
Deferred-W	Intra+H+O	1.336×	1.338×
CSR event scan	H	1.149×	1.153×
Compact states	H	1.398×	1.402×
Tier scheduling	H	1.029×	1.030×
Mapped O	O	1.239×	1.226×
Compact + mapped	Intra+H+O	1.226×	1.224×

- Geometric means across 4K–128K; each row has its own reference.
- Same masks and bitwise-identical last-token prefill logits on timing inputs.
- Isolated ratios should not be multiplied; separate Selector is excluded.

Paper, Sec. 3.6, PDF pp. 18–19; Table 6, p. 19.

Compact execution reduces complete-model peak allocation



- At 128K, both models save 26.222 GB of absolute peak memory.
- Sparse completes at 256K; Native OOMs during its first warmup.

Paper, Fig. 11 and Table 7, PDF p. 20. BF16, batch 1, cache enabled, last-token logits. OOM crosses are markers.

SparseGDN couples memory selection with the execution layout

1. Approximate local Delta solves, then spend recurrent budgets on input-dependent head-pair–chunk events.
2. Preserve gap decay and group-local reads while storing fewer state snapshots and constructing fewer correction matrices.
3. At about 29% keep, retain near-baseline aggregate RULER scores with 1.50×–1.94× GDN-core prefill speedup.

What further deployment evidence needs to resolve

Hand-designed proxies may miss rare transitions. Broader generation workloads, full-model latency, and explicit accelerator details matter.

Paper and version Publication and public code not found as of 2026-11-03.

Paper, Sec. 4, PDF p. 20; Sec. 5, p. 21.

Backup: the Delta solve produces two distinct right-hand sides

$$M = I + A_\Delta, \quad A_\Delta[t,s] = \beta_t \langle K[t], K[s] \rangle 2^{g_t - g_s} \mathbb{I}[s < t]$$

$$U \approx C\,\mathrm{diag}(\beta)V, \qquad W \approx C\,\mathrm{diag}(\beta \odot 2^g)K$$

$$A_c[i,j] = \langle Q_c[i], K_c[j] \rangle 2^{g_{c,i} - g_{c,j}} \mathbb{I}[j \le i]$$

- β enters both M and the right-hand sides.
- A_Δ uses key–key interactions for the local solve.
- A_c uses query–key interactions for output reconstruction.
- Four triangular blocks contribute 480 off-diagonal entries; three adjacent blocks contribute 768.

Paper, Secs. 2.1 and 2.3, PDF pp. 4–5. Entry-count decomposition is a teaching calculation.

Backup: tier counts target 30% retention

$$B(x,k) = \frac{k(n-k)}{n}(\overline{x}_{1:k} - \overline{x}_{k+1:n})^2$$

High-tier count n_H	Candidate medium counts n_M
1	11, 12, 13
2	7, 8, 9
3	3, 4, 5
4	0

$$\rho_{\text{nominal}} = \frac{n_H\rho_H + n_M\rho_M + (16 - n_H - n_M)\rho_L}{16}$$

Paper, Sec. 2.4.3, PDF p. 8. First/last valid chunks are anchors; keep counts use ties-to-even rounding.

Backup: gate visibility before recurrence

$$\eta_h = \frac{1}{4C} \sum_{c=0}^{C-1} \sum_{a=0}^{3} \left[\nu_{v_{c,a}} \, \mathrm{SiLU}(Z_{t_{c,a},h,v_{c,a}}) \right]^2$$

$$t_{c,a} = 64c + 8 + 16a, \qquad v_{c,a} = 73(4c + a) \bmod d_v$$

- Z is the pre-activation output gate projected from layer input.
- ν is the learned RMSNorm scale; out-of-range samples are zero.
- Sample one token per 16-token block and one rotating coordinate.
- Normalize gate energy relative to all value heads in the layer.

Paper, Sec. 2.4, PDF pp. 7–8. The selector factor is distinct from the model's output gate itself.

Backup: group-local output differs from a naive sparse recurrence

$$A_{c+1,c}[i,j] = \langle Q_{c+1}[i], K_c[j] \rangle 2^{g_{c+1,i}+\gamma_c-g_{c,j}}$$

$$S_{c+1} = 2^{\gamma_c} S_c + z_c K_c^\top (U'_c \odot 2^{\gamma_c - g_c})$$

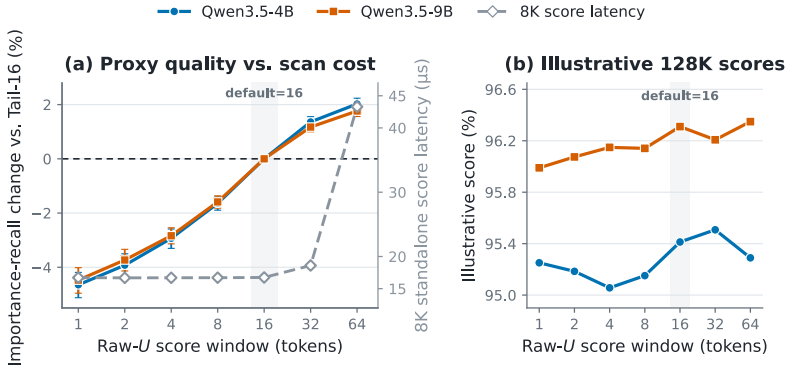
$$U'_{c+1} = \begin{cases} U_{c+1} - W_{c+1} S_{c+1}, & z_{c+1} = 1, \\ U_{c+1}, & z_{c+1} = 0. \end{cases}$$

Example: skip chunk 4, keep chunk 5

$S_5 = 2^{\gamma_4} S_4$ and $U'_5 = U_5 - W_5 S_5$, while O_5 still includes $A_{5,4} U_4 / \sqrt{d}$.

Paper, Sec. 2.7, PDF p. 11, Eqs. (24)–(25). The extra local read is not separately ablated for quality.

Backup: Tail-16 balances proxy recall with score cost



- Tail-64 improves recall, with 2.59× standalone scan cost.
- Production Tail-16 is fused into Intra's final on-chip tile.

Paper, Fig. 10, PDF p. 18; Sec. 3.5, pp. 17–18. Left differences: pp. Right scores are illustrative.

Backup: selector diagnostics differ in evidence strength

Change in percentage points	4B	9B
Add lifetime: importance recall	$+32.037$	$+31.484$
Tail vs random: importance recall	$+14.171$	$+13.646$
Remove visibility: RULER score	-0.487	-0.410

- Recall: 130 stratified 8K requests/model, visibility disabled.
- Tail vs random fixes tiers and exact budgets.
- Visibility: full 8K set, 1,300 examples/model; model gate unchanged.

Visibility-removal 95% CIs: $[-1.128, +0.128]$ and $[-1.077, +0.205]$ pp; both include zero.

Paper, Sec. 3.5, PDF p. 16. Dense-decode importance is a diagnostic proxy.

Backup: OOM entries have two distinct meanings

Model	Batch	64K	128K
4B	2	1.945×	1.988×
4B	4	1.833×	OOM†
4B	8	OOM†	OOM
9B	2	1.931×	1.939×
9B	4	1.830×	OOM
9B	8	OOM	OOM

- OOM†: Native OOM; Sparse independently completes.
- 4B batch-4/128K: 843.411 ms; batch-8/64K: 835.367 ms.
- Plain OOM: Sparse completion not independently verified.

Paper, Table 2 and Sec. 3.3, PDF p. 14. Timings are for the GDN core of the whole batch.

Backup: peak versus incremental memory

$$M_{\text{incr}} = M_{\text{peak}} - M_{\text{entry}}$$

Model	Length	Native peak	Sparse peak	Incr. reduction
4B	128K	53.116	26.894	59.56%
9B	128K	64.066	37.844	57.97%
4B	256K	OOM	44.637	—
9B	256K	OOM	56.794	—

- Peaks are in GB; entry allocation includes persistent model weights.
- Reduction percentages refer to the forward increment.
- 256K establishes prefill feasibility; quality is tested through 128K.

Paper, Sec. 3.7, PDF pp. 19–20; Table 7, p. 20. One warmup, three measured forwards; maximum peak reported.