# REMOVING REDUNDANT COMPUTATION IN A BDH-GPU-DERIVED LANGUAGE MODEL: EXACT RECURRENT TRAINING AND CONDITIONAL NEURON SELECTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Dragon Hatchling (BDH-GPU) combines strict-past $Q{=}K$ linear attention with a very wide positive neuronal representation. We study Arm-A, a 17.4M-parameter BDH-GPU-derived language model, and ask which apparent redundancy is actually removable. For attention, an exact packed segmented scan preserves the dense operator while avoiding cross-block score materialization. A predeclared Blackwell search improves our original dense path by $1.42\times$; against the fastest variant in that predeclared dense family (read blocks $\leq 512$), recurrence is $1.52\times$ faster at $T{=}2048$. At a fixed 131,072 positions/update, the recurrent/dense advantage grows from $1.04\times$ at $T{=}512$ to $2.12\times$ at $T{=}4096$. Applying the same mature executor to canonical 25M BDH still leaves recurrence below dense parity ($0.964\times$), and a 96-update matched BF16 trajectory differs by only $7.05 \times 10^{-4}$ held-out NLL despite non-bitwise parameter drift. For neuronal sparsity, causal controls show that static-support damage is directional rather than magnitude-only: preserving dense direction at the static-support norm reduces $\Delta$NLL from 0.0906 to 0.00819 relative to the converse norm-matched intervention, with the same ordering in KL. A prospective low-rank selector that uses only pre-$D_y$ information beats the fixed support at all three predeclared ranks on both evaluation pools. Exact algebraic redundancy is therefore removable now; for neuronal computation, the tested fixed support is insufficient while input-dependent support is substantially more faithful.

## 1 INTRODUCTION

Parameter sharing can make a depth-recurrent model compact without making its execution cheap. BDH-GPU makes this tension explicit: its GPU-oriented formulation combines strict-past $Q{=}K$ linear attention with a high-dimensional positive neuronal space whose activations become sparse during training (Kosowski et al., 2025). These properties suggest two superficially similar opportunities—remove explicit attention-score work, and skip inactive neurons—but they require different evidence.

We study Arm-A, a 17.4M-parameter BDH-GPU-derived model that retains the $E, D_x, D_y$ computational motif and depth recurrence while adding a same-document coordinator and a two-layer writer. We do not claim that Arm-A is canonical BDH, nor that recurrent linear attention is new. Our question is narrower: *which apparent redundancy in a BDH-derived model can actually be removed while preserving the computation?*

Figure 1 summarizes the distinction. Attention admits an algebraic identity, so the relevant test is whether an exact recurrent realization remains useful against a strong dense implementation. Neuronal sparsity has no analogous identity, so concentration alone is insufficient: support must be tested causally. The resulting paper is therefore an execution-and-intervention study rather than a new accelerator architecture.

Our evidence supports three claims. **(1) Exact recurrent execution remains favorable against a substantially strengthened dense baseline and shows context-dependent execution scaling.** At $T{=}2048$ on RTX PRO 6000 Blackwell, the fastest exact dense implementation in the evaluated



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

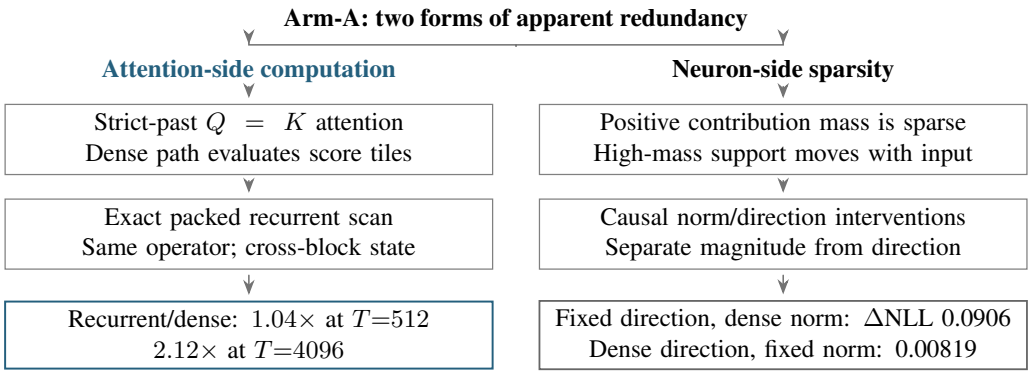


Figure 1: Two forms of apparent redundancy in Arm-A. Attention can be reformulated exactly and its advantage grows with context under a fixed position budget. Neuronal mass is concentrated, but static support can point in a damaging direction even after its magnitude is corrected.

predeclared family (RB$\leq$ 512) reaches 25.3k tokens/s, while recurrence reaches 38.4k tokens/s ($1.52\times$); with 131,072 positions/update held fixed, the ratio rises from $1.04\times$ at $T{=}512$ to $2.12\times$ at $T{=}4096$. The $T{=}4096$ point, which spans four $B{=}1024$ chunks, provides the strongest measured scaling evidence for recurrent handling of cross-chunk history. A 96-update BF16 matched trajectory keeps training and held-out losses closely aligned. **(2) Executor maturity alone does not explain the regime change.** Porting the same mature recurrent executor to canonical BDH raises its recurrent throughput by 3.9% but leaves it below dense parity ($0.964\times$). **(3) Static neuronal-support damage is not magnitude-only.** Norm/direction interventions measured with $\Delta$NLL and KL show substantially less damage when dense update direction is preserved than when static-support direction is preserved. A prospective pre-$D_y$ selector further predicts useful support without first constructing dense $y/u$, but we make no sparse wall-clock claim.

## 2 Setting and Related Work

### 2.1 BDH-GPU lineage and the evaluated model

Let $v_{t,\ell-1} \in \mathbb{R}^D$ be the model-space state of token $t$ entering recurrent depth step $\ell$. For each head, the evaluated BDH-derived model computes

$$x_{t,\ell} = \mathrm{ReLU}(D_x v_{t,\ell-1}), \qquad q_{t,\ell} = k_{t,\ell} = R_t x_{t,\ell}, \tag{1}$$

$$a_{t,\ell} = \sum_{\tau<t:\, s_\tau = s_t} (q_{t,\ell}^\top k_{\tau,\ell}) v_{\tau,\ell-1}, \qquad y_{t,\ell} = \mathrm{ReLU}(D_y\, \mathrm{LN}(a_{t,\ell}))\,, \tag{2}$$

$$u_{t,\ell} = x_{t,\ell} \odot y_{t,\ell}, \qquad b_{t,\ell} = \mathrm{LN}(E u_{t,\ell}). \tag{3}$$

Here $s_t$ identifies the packed document and $R_t$ is the rotary position transform (Su et al., 2024). The matrices $E, D_x, D_y$ are shared across all $L{=}8$ depth iterations. The repository extension adds

$$z_{t,\ell} = W_c v_{t,\ell-1} + b_c, \tag{4}$$

$$\mu^-_{t,\ell} = \begin{cases} 0, & t \text{ begins a document segment,} \\ \mathrm{mean}_{\tau<t:s_\tau=s_t}\, z_{\tau,\ell}, & \text{otherwise,} \end{cases} \tag{5}$$

$$c_{t,\ell} = \mu^-_{t,\ell} - z_{t,\ell}, \qquad g_{t,\ell} = 1 + \sigma(\alpha)\tanh(c_{t,\ell}), \tag{6}$$

$$\Delta_{t,\ell} = \mathrm{ReLU}((g_{t,\ell} \odot b_{t,\ell}) W_1)\, W_2, \qquad v_{t,\ell} = \mathrm{LN}(v_{t,\ell-1} + \Delta_{t,\ell}). \tag{7}$$

The writer width is 1040. Because these coordinator/writer and packing semantics are not canonical BDH, we call the model *BDH-GPU-derived*. We use the repository name Arm-A throughout. Exactness is stated only against the pinned Arm-A executable; canonical BDH is used separately as a control.

### 2.2 EXECUTION AND SPARSITY PRECEDENTS

Standard Transformer attention materializes pairwise token interactions (Vaswani et al., 2017). Linear attention can instead use a recurrent matrix state (Katharopoulos et al., 2020), while chunked formulations such as Gated Linear Attention emphasize that algebraic recurrence and efficient parallel training are different questions (Yang et al., 2024). FlashAttention makes the analogous systems point for softmax attention: exactness does not determine the memory-I/O schedule (Dao et al., 2022). The original BDH work already notes that state-space execution can become preferable at sufficiently long contexts (Kosowski et al., 2025); our claim is therefore not that recurrence is newly available. RetNet exposes parallel, recurrent, and chunkwise-recurrent forms (Sun et al., 2023), while Mamba-2/SSD develops block algorithms for recurrent state-space computation (Dao and Gu, 2024). Our contribution is narrower: an exact packed realization, stronger-baseline evaluation, scan-geometry characterization, and full-model controls for a BDH-derived geometry with unusually wide keys ($K{=}4096$/head in Arm-A; $8192$/head in the canonical control) and $D{=}256$ values. We have not established semantics-preserving adaptations of these neighboring systems to the same $Q{=}K$ operator, packing/reset contract, and surrounding Arm-A computation, so our timing claims are restricted to the evaluated implementation family rather than cross-model wall-clock comparisons. Recent BDH-CQ studies recurrent latent reasoning in the broader BDH family (Engdahl et al., 2026); our experiments concern language-model execution and make no reasoning or in-context-learning claim.

The neuronal result sits within contextual activation sparsity rather than generic pruning. Deja Vu showed that useful activation support is input-dependent and can be predicted ahead of expensive computation (Liu et al., 2023); the Lazy Neuron study characterized emergent activation sparsity in Transformers (Li et al., 2023). TEAL and CATS likewise exploit input-dependent activation sparsity or thresholding (Liu et al., 2025; Lee et al., 2024), while PowerInfer uses hot/cold neuron structure for serving (Song et al., 2024). These precedents make a static high-mass support an insufficient baseline by itself. We therefore distinguish concentration from causal fidelity, and separately evaluate a prospective selector without claiming a sparse executor speedup.

## 3 EXACT PACKED RECURRENT EXECUTION

### 3.1 DENSE SCORES AND THE EXACT STATE IDENTITY

For one head and one depth step, write $r_t = R_t x_t \in \mathbb{R}^K$ and $v_t \in \mathbb{R}^D$. Dense strict-past same-document attention is

$$a_t = \sum_{\tau<t:\, s_\tau=s_t} (r_t^\top r_\tau) v_\tau. \tag{8}$$

Define the rank-one update and exclusive same-segment state

$$P_\tau = v_\tau r_\tau^\top \in \mathbb{R}^{D\times K}, \qquad S_t = \sum_{\tau<t:\, s_\tau=s_t} P_\tau. \tag{9}$$

Then

$$S_t r_t = \sum_{\tau<t:\, s_\tau=s_t} v_\tau (r_\tau^\top r_t) = a_t, \tag{10}$$

so Eqs. 8–10 are identical in exact arithmetic. An ideal token recurrence performs state updates and state-vector products in $O(TKD)$ work with $O(KD)$ recurrent state. Our *implemented* training scan is chunked rather than token-serial: for block width $B$, it additionally forms a strict-lower local $B\times B$ score tile in each chunk. Its leading work is therefore $O(TKD + TB(K{+}D))$, with temporary storage $O(KD + B^2)$ per head up to batching and autograd. We report measured end-to-end training performance rather than treating the ideal $O(TKD)$ expression as the implemented FLOP count.

### 3.2 SEGMENTED CHUNKED SCAN

A token-by-token recurrence would destroy training parallelism. The evaluated path uses $B{=}1024$ chunks. Inside each chunk it computes local $QQ^\top$ scores, masks them to strict-past same-document interactions, and multiplies by local values; a $K \times D$ state carries only valid history from earlier chunks. At a packed-document boundary, the carried state is reset before the new segment contributes.

Table 1: Representative validation against the corresponding dense reference. Relative $L_2$ is shown in parentheses where available.

| Comparison | forward max | gradient max |
|---|---|---|
| FP64 dense/scan operator oracle | $1.42 \times 10^{-14}$ | – |
| Strong dense RB512 vs. original dense | 0 | $2.38 \times 10^{-7}$ |
| Recurrent vs. original dense | $1.43 \times 10^{-5}$ ($2.01 \times 10^{-7}$) | $2.56 \times 10^{-4}$ ($1.28 \times 10^{-4}$) |
| Canonical mature recurrent vs. dense | $1.53 \times 10^{-5}$ ($3.85 \times 10^{-7}$) | $4.00 \times 10^{-5}$ ($3.18 \times 10^{-5}$) |
| Packed previous-doc $\rightarrow$ next-doc leakage | 0 | – |

The implementation also uses branch-free packed-state updates, skips the provably zero initial carry, constructs product layouts directly, and caches RoPE phases. These transformations preserve Eqs. 1–7.

Packing is part of the operator contract. A prototype that flattened row identity together with segment identity could pass single-row tests while leaking attention across packed rows at microbatch $> 1$. We therefore require same-document isolation and packed-boundary parity before timing any path.

### 3.3 Numerical validation

Every timed path is checked against an exact reference before benchmarking. Table 1 reports representative full-model controls from the new baseline and same-executor experiments. TF32 is disabled for FP32 checks; BF16 is used only after these comparisons pass. The packed isolation test perturbs a preceding document and verifies that the following document is unchanged. Because mixed-precision accumulation order is not bitwise invariant, we additionally compare dense and recurrent BF16 optimization trajectories below rather than treating FP32 operator parity as a training-quality result.

The FP64 operator oracle confirms the algebraic rewrite to near machine precision. The larger FP32 gradient discrepancy relative to the forward discrepancy is consistent with finite-precision accumulation-order amplification through long reductions and eight shared depth iterations; we do not treat this as a full-model FP64 gradient certificate.

## 4 Exact Recurrence Under Strong Baselines and Scaling

### 4.1 A predeclared exact-dense baseline family

The central systems comparison uses the same trained Arm-A checkpoint, $T{=}2048$, BF16 autocast with FP32 master parameters, fused AdamW, and 131,072 positions/update on one RTX PRO 6000 Blackwell. The historical dense reference processed query blocks against their prefix and then masked invalid causal/document entries. To test whether its implementation quality explained the recurrent advantage, we predeclared exact dense variants with read blocks $B_r \in \{32, 64, 128, 256, 512\}$ plus a document-segmented implementation, and selected the fastest variant that fit the historical MB16$\times$4 geometry. This is a stronger exact baseline search, not a claim of a globally optimal dense kernel: the evaluated family stops at RB512, while dense RB1024/RB2048 were not tested. We therefore scope the $1.52\times$ result to this predeclared family rather than treating it as a global dense-vs-recurrent bound. The analyzed checkpoint is a 2.500B-token prefix of the frozen 5.0B-token `phase_bdh_stage2_5b_v1` stream, a deterministic whole-document mixture of 50% web, 20% StarCoder, 10% OpenWebMath, 10% algebraic mathematics, and 10% science; exact tokenizer/corpus hashes and packing manifests are in the supplement.

At MB16, RB32/64/128 exceed the memory budget; RB256, RB512, the segmented variant, the original reference, and recurrence fit. RB512 is the fastest exact dense path. The document-segmented implementation computes fewer invalid score products but is much slower because it decomposes work into many small GEMMs and host-side segment loops; this is why arithmetic elimination alone is not treated as performance evidence.

**Arithmetic count is not the performance ordering.** On the actual packed benchmark batch, 401.45M score pairs are valid after accounting for 64 rows and four heads. The original RB256 path

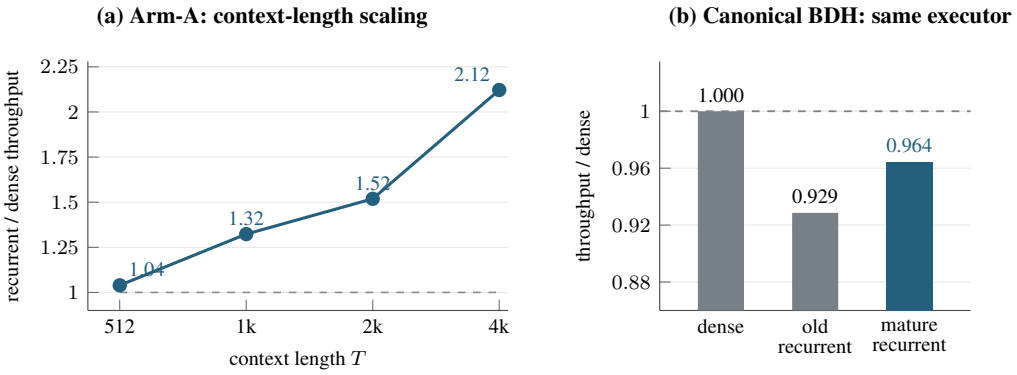


Figure 2: Systems controls. Left: with 131,072 positions/update held fixed, recurrent/dense throughput rises from 1.04× at $T$=512 to 2.12× at $T$=4096 against the predeclared RB512 dense comparator. Right: applying the same mature recurrent executor to canonical 25M BDH improves its recurrent path but leaves it below dense parity. The $T$=4096 point is a repacked systems workload, not a long-context quality claim.

Table 2: Blackwell $T$=2048 full-update comparison at the frozen Arm-A MB16×4 geometry. Ratios use median update time; peak is allocated CUDA memory.

| Path | tok/s | median ms | peak GiB |
|---|---|---|---|
| Original dense reference | 17,831 | 7350.6 | 83.98 |
| Strongest exact dense (RB512) | 25,290 | 5182.8 | 67.24 |
| Exact recurrent scan | 38,429 | 3410.8 | 63.18 |

evaluates 603.98M score entries, the document-segmented variant 466.63M, and RB512 671.09M, yet RB512 is much faster than both. The recurrent $B$=1024 path likewise forms local chunk score tiles and uses its $K \times D$ state only for cross-chunk history. Large-GEMM efficiency, state traffic, temporary storage, and host launch structure therefore matter more than scalar score count alone.

The stronger dense implementation is 1.418× faster than the original dense path, reducing the apparent recurrent advantage from roughly 2.15× to 1.520×. Recurrent peak allocation is 6.0% below RB512 (63.18 versus 67.24 GiB); the larger 24–25% reduction applies only relative to the original dense reference.

### 4.2 Context scaling and scan geometry

Holding 131,072 positions/update fixed, RB512 versus recurrent throughput is 43.3k/45.0k at $T$=512 (1.040×), 35.4k/46.9k at $T$=1024 (1.323×), 25.3k/38.4k at $T$=2048 (1.519×), and 15.8k/33.4k at $T$=4096 (2.122×). RB256 is slower than RB512 wherever both fit; at $T$=4096 RB256 does not fit. Because the production scan block is $B$=1024, the $T$=512 and $T$=1024 workloads fit within a single local chunk and exercise no cross-chunk recurrent carry; their measured gains reflect local tiling and executor structure rather than elimination of cross-chunk score work. Only $T > B$ exercises recurrent handling of cross-chunk history. We therefore interpret the sweep as context-dependent execution scaling, not as a bracketed crossover or as evidence that state carry itself causes every measured gain. In particular, $T$=4096 spans four $B$=1024 chunks and is the strongest point in this sweep for the cross-chunk state-carry regime. The 4096-token workload pairs adjacent frozen 2048-token rows, preserving a cross-row segment only when token and position metadata prove document continuation; it therefore tests execution scaling, not long-context language quality. MB16 fits through $T$=2048; both primary paths use the largest common MB8 geometry at $T$=4096.

The recurrence is also materially sensitive to chunk geometry. At $T$=2048 and fixed MB16×4, scan blocks 256/512/1024 achieve 28.1k/32.4k/38.5k tokens/s, with peak allocations 73.68/68.19/63.18 GiB. Thus the production block 1024 is 37.2% faster than block 256 while using 10.5 GiB less peak allocation. This supports a systems conclusion rather than an algebraic one: exact recurrence exposes an opportunity, but chunk geometry strongly controls whether the opportunity is realized.

### 4.3 SAME EXECUTOR ON CANONICAL BDH

A second control asks whether the earlier canonical-BDH negative result was merely an implementation-quality artifact. We keep the public 25M BDH architecture unchanged ($D{=}256$, $H{=}4$, $K{=}8192$/head, eight levels) and replace only its recurrent attention execution with the same mature chunked executor used above. Because this is an execution-throughput control rather than a language-quality evaluation, it uses deterministic byte-token tensors and identical weights, optimizer state, RNG, $T{=}2048$, and MB1×8 geometry across its dense/recurrent paths. This geometry is not matched to Arm-A's MB16×4 contract, so the $0.964\times$ result rules out executor maturity alone under the evaluated canonical geometry but does not isolate architecture from batch/geometry effects.

Canonical dense reaches 16,567 tokens/s. The prior recurrent realization reaches 15,384 tokens/s ($0.929\times$ dense), while the mature executor reaches 15,977 tokens/s: a 3.9% recurrent improvement, but still only $0.964\times$ **dense**. Thus executor maturity alone does not explain why recurrence is favorable in Arm-A. This experiment does *not* identify the causal architectural factor; neuronal width, packing, coordinator/writer structure, and other model differences remain entangled.

### 4.4 SHORT-HORIZON BF16 TRAINING CONSISTENCY

Exact FP32 operators need not produce bitwise-identical BF16 optimization trajectories. We therefore clone one trained checkpoint and optimizer state into dense RB512 and recurrent arms, replay identical real packed batches with paired RNG states, and run 96 optimizer updates (12.58M tokens/arm) at MB16×4. Mean absolute training-loss gap is $7.57 \times 10^{-4}$ (maximum $5.22 \times 10^{-3}$). Final held-out NLL is 2.861159 for dense and 2.860455 for recurrence, a gap of $7.05 \times 10^{-4}$. The parameter vectors nevertheless separate to 2.92% relative $L_2$, consistent with different BF16 accumulation orders. We do not provide a dense-vs-dense numerical noise floor or normalize this separation by total displacement from the starting parameters; accordingly, we interpret the experiment only as short-horizon loss consistency, not numerical identity, parameter-trajectory equivalence, or convergence equivalence.

### 4.5 PRODUCTION RUN AND PORTABILITY SCOPE

The systems controls above use matched eager benchmarks. Separately, an optimized recurrent production configuration completes 19,074 updates and 2,500,067,328 training tokens in 8.69 h, with saved production telemetry reporting approximately 80.4k tokens/s and 58.20 GiB peak allocation. Before the long run, the Blackwell compiled path was separately qualified: full-model FP32 discrepancy was at most $6 \times 10^{-8}$, the BF16 path discrepancy was $9.8 \times 10^{-4}$, compiled full-update repeatability differed by $9.31 \times 10^{-10}$, and the harness reported zero graph breaks. We do not divide the production rate by the matched eager baselines because compilation and geometry differ; it establishes sustained trainability rather than a matched speed ratio.

Portability certificates against the original L4/Ada and RTX A2000/Ampere references are retained in the supplement, but the stronger dense-baseline search was not repeated on those GPUs. Recurrent inference is stateful with 128.008 MiB FP32 state in the validated configuration; recurrent and full-prefix logits agree to $1.31 \times 10^{-6}$ with token-identical greedy generation through the 2048-token training-window policy.

## 5 CONDITIONAL NEURONAL SUPPORT

### 5.1 CONCENTRATION DOES NOT IDENTIFY A FAITHFUL STATIC DIRECTION

The neuronal contribution is nonnegative,

$$u = x \odot y, \qquad m(A; u) = \frac{\sum_{i \in A} u_i}{\sum_i u_i}. \tag{11}$$

Across 32 layer-head cells, per-token top-64 coordinates retain 0.7837 of contribution mass, while a globally fixed top-64 set retains 0.3622. The local-global gap remains 0.419–0.424 over tested contexts $T \in \{256, 512, 1024, 2048\}$, confirming that high-mass support moves with the input.

The more important question is causal. Our frozen norm/direction intervention evaluates one width-320 static support on two natural-text pools and all eight depths using prediction-relevant metrics.

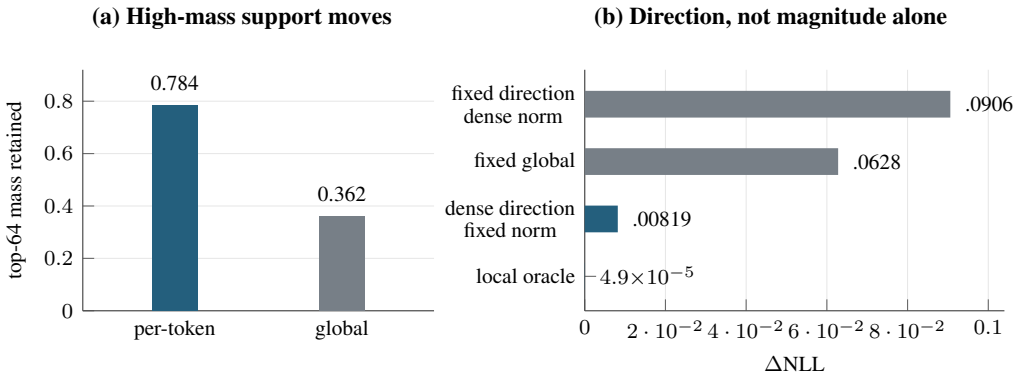


Figure 3: Left: contribution mass is strongly input-dependent. Right: causal width-320 interventions measured by ΔNLL. Restoring magnitude does not rescue the static-support direction; preserving dense direction at the smaller norm sharply reduces damage.

Table 3: Frozen width-320 intervention reanalysis aggregated by evaluation row across depths. Lower ΔNLL/KL is better. These artifacts do not contain a retained-mass grid, so this table does not make an equal-mass claim.

| Intervention | ΔNLL | KL |
|---|---|---|
| Fixed global support | 0.062794 | 0.066653 |
| Fixed direction, dense norm | 0.090595 | 0.091235 |
| Dense direction, fixed norm | 0.008192 | 0.006665 |
| Local oracle, width 320 | 0.000049 | 0.000037 |

For dense logits $\ell$ and intervention logits $\tilde{\ell}$, we report target ΔNLL and forward $D_{\mathrm{KL}}(p_{\mathrm{dense}} \| p_{\mathrm{int}})$; relative logit $L_2$ is retained only as a secondary diagnostic. The support mask is applied to $u$ at one depth $d$, then the writer output $\Delta_{t,d}$ at that same depth is used for the direction/magnitude control. Norms are per token over the model dimension. FIXED-NORM-MATCHED preserves the fixed-support writer-update direction and rescales it by $\|\Delta^{\mathrm{dense}}\|_2 / \max(\|\Delta^{\mathrm{fixed}}\|_2, \epsilon)$; DENSE-DIRECTION-FIXED-NORM keeps the dense direction and rescales it by the reciprocal norm ratio. Other depths remain unmodified in that intervention.

The result is not explained by a simple loss of magnitude. Restoring dense norm while retaining the static-support direction actually increases damage, whereas retaining dense direction at the smaller fixed-support norm reduces ΔNLL by nearly an order of magnitude relative to the fixed intervention. A paired 10,000-resample bootstrap clustered by evaluation row gives fixed-minus-dense-direction differences of 0.0546 in ΔNLL (95% CI $[0.0454, 0.0636]$) and 0.0600 in KL ($[0.0520, 0.0682]$). The more specific norm-matched direction comparison is also positive independently on NATURAL_A ($[0.0369, 0.0603]$) and NATURAL_B ($[0.0824, 0.1225]$). We therefore claim that static-mask damage is *partly directional, not magnitude-only*. We do not claim that dynamic support is superior at equal retained neuronal mass because the frozen artifacts contain no suitable mass-by-width grid. A zero-$u$ reference existed only in the earlier logit-perturbation protocol; the frozen ΔNLL/KL reanalysis did not include a matched zero-$u$ arm, so we do not mix that older row into Table 3.

### 5.2 PROSPECTIVE SELECTION AND LIMITS

The contextual-sparsity literature makes a prospective selector the relevant next step (Liu et al., 2023; Lee et al., 2024). We therefore evaluate a deterministic low-rank predictor before $D_y$. To match the implementation exactly, define $\bar{a}_{t,h} = \mathrm{LN}(a_{t,h})$ and let $\widetilde{D}_y^{(h)} = (D_y^{(h)})^\top \in \mathbb{R}^{D \times K}$ denote the row-vector storage convention used by the code. For each head, the implementation computes a truncated SVD $\widetilde{D}_y^{(h)} \approx U_r^{(h)} V_r^{(h)}$, with singular values folded into $U_r$. The captured selector input called `a` in the evaluator is already this layer-normalized attention output $\bar{a}$. Using only $\bar{a}_{t,h}$, the

already-available positive vector $x_{t,h}$, and frozen $D_y$ weights, it forms

$$\hat{y}_{t,h} = \mathrm{ReLU}\Big(\bar{a}_{t,h} U_r^{(h)} V_r^{(h)}\Big), \qquad s_{t,h} = x_{t,h} \odot \hat{y}_{t,h}, \qquad A_{t,h} = \mathrm{TopK}_{320}(s_{t,h}). \tag{12}$$

No dense $y$ or $u$ is constructed before selecting $A_{t,h}$. "Rank" is the SVD rank $r \in \{16, 32, 64\}$, not the number of selected coordinates. The fixed baseline is calibrated separately by summing $u$ mass over the calibration pack and freezing the global top-320 support; evaluation then uses the disjoint frozen evaluation pack. At $T$=2048, both natural-text pools, and all eight depths, every predeclared rank improves $\Delta$NLL over the fixed support under the clustered-by-evaluation-row criterion: each rank has a 95% improvement interval above zero in both pools and satisfies the predeclared depth-consistency gate. Exact per-rank $\Delta$NLL values, retained-mass/overlap diagnostics, depth-wise results, and clustered intervals are reported in the supplement. At rank 64, the low-rank projection itself costs about $64(256{+}4096) = 278{,}528$ multiply-accumulates per token/head, versus $256 \times 4096 = 1{,}048{,}576$ for dense $D_y$ (about 26.6% before top-$k$ and other overhead). This is an operation-count motivation, not a wall-clock result: the predictor is not a learned router, its selection overhead is not yet fused into a sparse kernel, and we make no end-to-end sparse speedup claim.

## 6 Discussion and Limitations

The strongest conclusion is narrower than an "accelerator-aware architecture" claim. Exact recurrence remains favorable after the dense baseline is strengthened substantially; its measured advantage grows with context; and a matched BF16 trajectory remains loss-aligned over 12.58M tokens/arm. At the same time, porting the mature executor to canonical BDH leaves recurrence below dense parity. These controls rule out the original dense reference's weakness as the sole explanation, a single-$T$ artifact, and executor maturity as complete explanations under the evaluated geometries, but they do not establish a globally optimal dense comparator or identify one causal architectural factor. In particular, dense RB1024/RB2048 remain unevaluated, and the canonical control uses a smaller MB1$\times$8 contract than Arm-A. A factorial study over key width, packing, coordinator/writer structure, and executor choices would be needed for attribution.

The implemented scan is not the ideal $O(TKD)$ token recurrence: its chunks retain local quadratic work, and the scan-block sweep shows that execution geometry materially changes throughput and memory. The model is 17.4M parameters, the strongest-baseline and scaling studies are Blackwell-only, and Hopper was not tested; we make no model-size scaling claim or exhaustive GPU-generalization claim. The 4096-token point is a repacked execution workload, not evidence of 4096-context language quality. The 96-update BF16 experiment supports short-horizon loss consistency rather than convergence equivalence.

The neuronal evidence is deliberately bounded. $\Delta$NLL and KL, rather than logit $L_2$ alone, support the direction-versus-magnitude conclusion, but retained-mass matching is unavailable in the frozen norm/direction artifacts. The prospective selector is evaluated at one width/context and three predeclared ranks, and no sparse executor is timed. Thus the paper establishes that input-dependent support is substantially more faithful than the tested fixed support, not that every static sparsification must fail or that deployable sparse acceleration has already been achieved.

## 7 Conclusion

We studied two forms of apparent redundancy in a BDH-GPU-derived language model. Exact recurrence is the removable kind algebraically: after a predeclared search strengthens the original dense baseline by 1.42$\times$, recurrence is 1.52$\times$ faster than the fastest tested dense variant (RB$\leq$ 512) at $T$=2048, and the measured ratio grows to 2.12$\times$ at $T$=4096 under a fixed position budget. The four-chunk $T$=4096 point provides the strongest measured evidence for the cross-chunk recurrent regime. Scan-block sensitivity and the canonical same-executor control show that systems realization matters, while a 96-update BF16 trajectory remains loss-aligned despite non-bitwise parameter drift. Neuronal sparsity is different: causal norm/direction controls show that a static support can induce a damaging update direction even when magnitude is restored, while prospective input-dependent selection is substantially more faithful. Exact algebraic redundancy can therefore become an execution optimization now; for neuronal sparsity, our results motivate input-dependent support selection rather than treating a high-mass fixed support as removable computation.

### AI USE STATEMENT

Generative AI tools assisted literature retrieval and synthesis, research planning, implementation and debugging, experiment orchestration, derivation checking, analysis of experimental outputs, manuscript drafting and editing, and figure/table/reference formatting. All quantitative claims were produced by executable experiments and checked against saved artifacts, benchmark logs, source inspection, and numerical-correctness tests. The human authors set the research questions, experimental contracts and falsification criteria, chose baselines and interventions, operated the reported compute, adjudicated failed and valid runs, and reviewed the final manuscript. AI was not used to generate the evaluation datasets. The authors take responsibility for the experimental design, executed results, code, analyses, citations, and conclusions.

### REPRODUCIBILITY STATEMENT

The main text specifies model shape, packing/reset semantics, the dense-to-recurrent identity, optimizer/precision contract, benchmark token geometry, scan-block sweep, context-length sweep, BF16 trajectory protocol, correctness tests, intervention definitions, and hardware classes. The anonymous supplement contains pinned executable source, source/checkpoint hashes where applicable, raw timing samples, dense/recurrent correctness and boundary tests, production metadata, analysis scripts, and a claim-to-artifact ledger. Canonical BDH is pinned to upstream commit `2b0d7a45...` and the public 25M checkpoint revision `d8895c2f....` Failed or blocked production attempts and invalidated sparse measurements are retained in the audit trail and are not used as results.

## REFERENCES


Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 10041–10071. PMLR, 2024.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, 2022.

Bjorn Engdahl, Adrian Kosowski, Jan Chorowski, Zuzanna Stamirowska, Przemyslaw Uznanski, Junlin Jiang, Rohan Phadke, Remigiusz Kinas, and Richard Zhong. BDH-CQ: In-context learning with recurrent latent reasoning. *arXiv preprint arXiv:2608.09888*, 2026.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Adrian Kosowski, Przemyslaw Uznanski, Jan Chorowski, Zuzanna Stamirowska, and Michal Bartoszkiewicz. The dragon hatchling: The missing link between the transformer and models of the brain. *arXiv preprint arXiv:2509.26507*, 2025.

Donghyun Lee, Je-Yong Lee, Genghan Zhang, Mo Tiwari, and Azalia Mirhoseini. CATS: Contextually-aware thresholding for sparsity in large language models. In *Conference on Language Modeling*, 2024.

Zonglin Li, Chong You, Srinadh Bhojanapalli, Daliang Li, Ankit Singh Rawat, Sashank J. Reddi, Ke Ye, Felix Chern, Felix Yu, Ruiqi Guo, and Sanjiv Kumar. The lazy neuron phenomenon: On emergence of activation sparsity in transformers. In *International Conference on Learning Representations*, 2023.

James Liu, Pragaash Ponnusamy, Tianle Cai, Han Guo, Yoon Kim, and Ben Athiwaratkun. Training-free activation sparsity in large language models. In *International Conference on Learning Representations*, 2025.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, and Beidi Chen. Deja vu: Contextual sparsity for efficient LLMs at inference time. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 22137–22176. PMLR, 2023.

Yixin Song, Zeyu Mi, Haotong Xie, and Haibo Chen. PowerInfer: Fast large language model serving with a consumer-grade GPU. In *Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles*, pages 590–606, 2024. doi: 10.1145/3694715.3695964.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.