

Streaming Is Not Memory:

Constant-State Born-Attention Byte Models on PG-19

Marcel Crasmaru
crasmarum@gmail.com

October 22, 2026

Abstract

A language model whose attention keeps a fixed-size state can read an input of any length in one pass, with memory that does not grow with the input. That property is often reported as a long context. We separate it into three claims and test each one on byte-level PG-19. The models are fully complex-valued, built in the CNet framework, and use linear Born attention, $|\langle q, k \rangle|^2$ evaluated through a density-matrix state, with a fixed per-head decay and rotary positions. A model with 12M complex parameters (23.9M real-scalar equivalents) trained in strict FP32 on half a pass of the full 11.7 GB corpus scores 1.312 bits per byte on the PG-19 test partition when each 5,242,880-byte window is read in one unbroken pass with 12.8 MB of state; the MBLM hierarchy reports 2.448 at that window length, an uncontrolled comparison, and 1.240 at 98k bytes. Streaming is stable, but we find little aggregate benefit from persistence beyond a few thousand bytes. A reset sweep on identical validation bytes shows that carried state helps by 0.012 bits per byte for the 12M model and 0.006 for a 4M model, and both curves follow a fixed-cost-per-reset model, $\Delta L(C) = \Delta L(\infty)(1 - 4096/C)$, with residuals below 4×10^{-5} . This is consistent with most of the gain coming from state recovery after a reset. Extending the reset interval beyond 256k bytes improves the loss by about 1.8×10^{-4} bits per byte, as the fixed-cost model predicts; the residual is below 4×10^{-5} , but an aggregate-loss experiment does not isolate the contribution of long-range information, and retrieval was not measured. The fixed decay suppresses old contributions exponentially, with half-lives of 22 to 709 bytes per head. A real-valued Transformer with the same parameter count, bytes and schedule, scored on the same windows, reaches 1.245 bits per byte on test against 1.324 for the Born model with the same 4,096-byte resets. In a matched five-configuration experiment at 10,000 steps in PyTorch, the same complex Born design achieves the lowest test loss, 1.413 against 1.495 for the Transformer, while Born attention placed in the real Transformer costs 0.017: the relative performance of Born and softmax depends on the representation and read-out around them. CNet’s default initialization, which we show to be poorly conditioned, accounts for 0.023 of the original gap: with a Gaussian start the same design reaches 1.301 after 60,000 steps, and the Transformer still leads at full length (1.245). The ranking depends on the training budget: the complex Born design is better with a 10,000-step schedule and worse with a 60,000-step one. Constant-state streaming is a property of the method; long effective memory is not; neither is quality at equal size at full training length.

1 Introduction

Attention layers with a finite feature map can be computed by a recurrence whose state has a fixed size [4, 8, 9]. A model built from such layers can process an arbitrarily long input in one pass, in time linear in its length and in memory that does not depend on it. Born attention [1], whose weights are $|\langle q, k \rangle|^2$ over complex queries and keys, is one such layer: the degree-two kernel has the exact feature map $x \mapsto xx^\dagger$, and rotary positions [7] survive the linearization exactly.

Results for such models are often summarized as a context length. That conflates three different statements:

1. *Streaming*: the model can process a window of length W with state of constant size.
2. *Quality preservation*: its loss does not deteriorate when state is carried across the whole window.
3. *Long-range use*: information from far back in the window improves its predictions.

The first is a structural property of the recurrence. The second is an empirical question about numerical stability and distribution shift. The third is the one usually implied by “context length”, and it is the one that a fixed-size state does not guarantee.

This paper tests all three on byte-level PG-19 [6] at the window length of the MBLM evaluation [3], 5,242,880 bytes. In one sentence: we demonstrate constant-state streaming over 5M-byte windows, but find that extending state persistence yields only a small improvement in average loss, while the comparison with a Transformer matched in real-scalar parameters depends on the budget: the complex Born design is better than the Transformer at 10,000 steps and worse at 60,000, also after correcting CNet’s poorly conditioned default initialization. Our contributions are:

- Two fully complex-valued byte models with linear Born attention and a fixed per-head decay (4M and 12M complex parameters, 8.1M and 23.9M real-scalar equivalents), trained in strict FP32 with complete compute accounting (§3).
- A carried-state evaluation that reads each 5M-byte test window in one pass, and its result against MBLM’s number for that window length (§4).
- A reset sweep on identical validation bytes that measures how much carried state helps as a function of the reset interval, for both models (§6).
- A real-valued Transformer baseline matched in parameters, bytes, vocabulary, schedule and scoring protocol, and a 10,000-step ablation that crosses real and complex streams with softmax and Born attention (§5).
- An analysis of decay along dependency paths: per-head half-lives, a proof that the explicit decay factor along any path through the stack is at most $\gamma_{\max}^{t-s}$, and an illustrative estimate of scale (§7).

2 Model

The models are built in CNet [2], a C++/CUDA framework in which every activation and weight is complex and gradients follow the Wirtinger calculus. Bytes are embedded as complex vectors of width E . Each of L pre-norm blocks applies a per-token normalization, multi-head Born attention with rotary positions, and a complex feed-forward layer of width $4E$, each inside a residual connection. A per-token projection and a Born-rule cross-entropy produce the next-byte distribution. There is no learned positional table, so nothing in the model is tied to the training sequence length.

Table 1: Fixed decay per head. Half-life is $\ln 2 / (-\ln \gamma_h)$ bytes. The last two columns give the weight γ_h^D of a contribution D bytes back, relative to the current byte.

head h	γ_h	half-life (bytes)	$D = 4,096$	$D = 16,384$	in model
0	0.96875	21.8	≈ 0	≈ 0	4M, 12M
1	0.984375	44.0	≈ 0	≈ 0	4M, 12M
2	0.9921875	88.4	1×10^{-14}	≈ 0	4M, 12M
3	0.99609375	177	1×10^{-7}	≈ 0	4M, 12M
4	0.998046875	355	3.3×10^{-4}	1×10^{-14}	12M
5	0.9990234375	709	1.8×10^{-2}	1×10^{-7}	12M

Linear Born attention with decay. For a head with complex queries, keys and values of width d , let $\phi(x) = \text{vec}(xx^\dagger) \in \mathbb{C}^{d^2}$, so that $\phi(q)^\dagger \phi(k) = |\langle q, k \rangle|^2$. With a fixed decay $\gamma_h \in (0, 1)$ per head, the output at position t is

$$S_t = \gamma_h S_{t-1} + \phi(k_t) v_t^\top, \quad z_t = \gamma_h z_{t-1} + \phi(k_t), \quad y_t = \frac{\phi(q_t)^\dagger S_t}{\phi(q_t)^\dagger z_t}, \quad (1)$$

which equals quadratic Born attention with weights $\gamma_h^{t-s} |\langle q_t, k_s \rangle|^2$ normalized over $s \leq t$. Here $v_t^\top$ is the plain transpose: values do not enter the kernel, and the implementation stores them as $d_v = 2d$ real numbers. Rotary positions multiply coordinate j of a query or key at position t by $e^{it\theta_j}$. In ϕ the diagonal entries $|k_j|^2$ are then position-invariant, while each off-diagonal entry $k_j \bar{k}_{j'}$ acquires the phase $e^{it(\theta_j - \theta_{j'})}$, so $\phi(q_t)^\dagger \phi(k_s)$ depends on position only through $t - s$ and the recurrence keeps rotary positions exactly. Training uses the exact chunkwise form of (1) with chunk 512; evaluation carries (S, z) from one 4,096-byte forward to the next. With values stored as $d_v = 2d$ real numbers, the state holds $d^2(d_v + 1)$ floats per head. For $d = 32$ that is 66,560 floats, so the 12M model below carries 1.60 MB per layer and 12.8 MB in total, whatever the length of the input.

Decay schedule. The decay follows the RetNet schedule [8], $\gamma_h = 1 - 2^{-(5+h)}$ for head $h = 0, \dots, H-1$. Table 1 lists the half-lives. A model with four heads stops at $h = 3$; a model with six heads reaches $h = 5$, whose half-life is four times longer.

3 Setup

Data. PG-19 [6] with its official splits: 28,602 training books (11,678,213,269 bytes), a validation split of 17,733,052 bytes and a test split of 41,289,201 bytes (100 books). The models read raw bytes; nothing decodes UTF-8, so a two-byte character is two tokens and “per character” in the training logs means per byte. A 256-entry table maps each byte value that occurs in the training and validation files to a class: 222 classes for the full corpus, 174 for the 1,200-book subset used by the 4M model. Test bytes outside a model’s classes are charged 8 bits each.

Models and training. Both models use the architecture of §2 with RoPE only, Adam, per-component gradient clipping at ± 0.5 on the gradient summed over the batch, a probability floor of 10^{-3} in the Born loss, and 4,096-byte training sequences. Table 2 gives the rest. Both train in strict FP32: no TF32 and no reduced-precision arithmetic.

The 12M model trains data-parallel over NCCL with 6 sequences per GPU. CNet allocates a static 63.8 GB arena per GPU once at start-up. Across 344 logged 100-step averages, step time

Table 2: Training runs. GPU-hours count every GPU of the node for the whole training time.

	4M model	12M model
complex parameters (real numbers)	4.03M (8.06M)	11,966,976 (23.9M)
width, blocks, heads $\times$ head width	256, 6, 4×32	384, 8, 6×32
training text	1,200 books, 472 MB	full corpus, 11.7 GB
batch $\times$ steps	$8 \times 15,000$	$24 \times 60,000$
bytes seen	0.49 GB	5.90 GB (half a pass)
learning rate	trainer default	10^{-3} , 2k warmup, cosine to 10^{-4}
hardware	$1 \times$ GH200	$4 \times$ H100 SXM5 80 GB, NVLink
step time	—	694.3 ms (sd 0.36 ms over 100-step means)
training time	2.4 h	11.6 h
GPU-hours	2.4	46.4
cost at on-demand prices	—	\$197 at \$17 per node-hour

stayed between 693.9 and 700.1 ms. These are averages and would hide single-step spikes; per-step variance, data-loader wait and scaling to fewer GPUs were not measured.

Evaluation protocols. Three measurements are reported, and they are not interchangeable.

- *Training-time validation:* every 2,000 steps, 20 batches of random 4,096-byte chunks of the validation split, about 0.49 MB per check, in nats per byte. It is noisy at the 0.01 to 0.02 level. CNet keeps the best check’s model and, at the end, replaces it with the final model if one more such check is at least as good; for the 12M run the final check scored 0.921 against a best of 0.936, so its checkpoint is the step-60,000 model.
- *Carried-state window score:* the file is cut into consecutive 5,242,880-byte windows. Each window is read as 1,280 consecutive 4,096-byte forwards with (S, z) carried between them and rotary positions at the absolute byte offset; state resets only at a window boundary. Bytes are scored across the 4,096-byte forward boundaries inside a window: the last position of each forward predicts the first byte of the next. Every byte of a window is scored except its first, which has no context. A partial final window is read in whole 4,096-byte blocks; the last position of the last block still predicts the byte after it, so of the remainder all but that byte goes unscored. On test the last window holds 4,589,041 bytes, 1,120 blocks plus 1,521 bytes, of which 1,520 are unscored, and the scored total is $7 \times 5,242,879 + 4,587,520 = 41,287,673$ bytes. This is the protocol of MBLM’s 5M configuration [3], in bits per byte. With 4,096-byte windows (“4k resets”), the same rule scores 4,095 bytes per window, 41,277,600 on test.
- *Reset sweep:* the carried-state score on the first 15,728,640 validation bytes (three full windows) with the state reset every C bytes, for C from 4,096 to 5,242,880. We report $\Delta L(C) = L(C) - L(4,096)$ in bits per byte; negative values mean carried state helps. C is how long state may persist, not an effective context length. Out-of-vocabulary bytes are charged identically at every C and cancel in ΔL .

4 Results

Training. Table 3 shows training-time validation for the 12M model. It crossed 1.0 nats per byte at step 32,000, with the learning rate still at half its peak. The 4M model’s best was 1.121 on its own, smaller vocabulary. Samples are best described as locally coherent literary continuation

Table 3: 12M model, training-time validation (nats per byte; bits per byte = nats / $\ln 2$).

step (k)	2	4	8	12	16	20	24	28	32	34
val	1.894	1.447	1.203	1.113	1.087	1.059	1.051	1.009	0.987	0.976

later checks: 0.955 (46k), 0.936 (52k), 0.949 (60k; end-of-run check 0.921, keeping the step-60,000 model)

Table 4: PG-19, carried-state window score in bits per byte. “4k resets” is the same file with the state reset every 4,096 bytes.

	MBLM, 5M context [3]	4M model	12M model
test, 5M windows	2.448	1.656	1.312
test, 4k resets	—	—	1.324
validation, 5M windows / 4k resets	—	1.690 / 1.696	1.355 / not run*
parameters (real numbers)	350M	8.06M	23.9M
training bytes	100 GB	0.49 GB	5.90 GB
state during evaluation	grows with context	constant	constant, 12.8 MB

*Whole-split 4k-reset score not run for the 12M model; on the first 15.7 MB of validation it is 1.371 with 4k resets against 1.359 with 5M windows (Tables 5 and 8). The 4M figures in this row are on the whole validation split.

with subsequent semantic collapse: correct quotation, speaker tags, register and period vocabulary, with meaning that drifts within one or two sentences (Appendix B). An exploratory comparison of checkpoints under GPT-2 is in Appendix B.

5M-byte windows. Table 4 gives the carried-state window scores. The 4M model scores 1.656 bits per byte on the test partition, between 1.60 and 1.79 per window. MBLM reports 2.448 for its 5M-byte configuration, a 350M-parameter hierarchy of a Mamba-2 stage over two Transformer stages with hidden size 256, trained on 100 GB on one A100 for 15 h [3]. This places a constant-state model next to the one published system evaluated at this window length; it is not a controlled comparison, since parameter count, data, hardware and architecture all differ. MBLM’s strongest PG-19 results are at shorter context (1.240 at 98k bytes, against 1.370 for their MegaByte [10] baseline), and our models do not reach those.

The validation row answers the first two claims for the 4M model. Streaming works: 1,280 chained forwards per window with no reset produce no drift, and the per-window test scores stay in a narrow band. Quality is preserved: 1.690 with the state carried against 1.696 with resets. The same row bears on the third claim. The carried state improves the score by 0.006 bits per byte, so information older than 4,096 bytes contributes little to the aggregate loss.

5 Matched Transformer baseline

To test whether the complex Born model is competitive at equal size, we trained a standard real-valued Transformer on the same bytes: a pre-norm decoder with 8 layers of width 512, 8 heads, feed-forward width 1,888, rotary positions and causal FlashAttention through PyTorch, 24,091,136 parameters against the Born model’s 23,933,952 real numbers. It uses the same 222-class byte table, 60,000 steps of $24 \times 4,096$ bytes, the same learning-rate schedule, Adam, and the same clipping: CNet clips each component of the gradient summed over the batch to ± 0.5 , which for Adam, scale-invariant up to its ϵ , would equal clipping the mean gradient at $0.5/24$ if the two gradients had the same units. They do not: CNet’s gradient is the Wirtinger derivative $\partial L / \partial \bar{z} = \frac{1}{2}(\partial L / \partial x + i \partial L / \partial y)$, half

Table 5: Matched baseline, bits per byte. “4k windows”: state or context reset every 4,096 bytes, first byte of each window unscored; identical byte sets for all models. All three trained 60,000 steps on the same schedule. The mirror is the PyTorch version of CNet’s model with a Gaussian initialization (§5, Table 7).

	CNet, default init	mirror, Gaussian init	Transformer
test, 4k windows	1.324	1.301	1.245
test, 5M windows, carried state	1.312	—	n/a
validation, first 15.7 MB, 4k windows	1.371	1.355	1.287
validation, whole split	1.355 (5M)	—	1.286 (4k)
parameters (real numbers)	23.9M	23.9M	24.1M
precision	FP32	bf16 linear, fp32 scores	bf16 autocast
time per step	694 ms (4×H100)	304 ms (8×A100)	39 ms (4×H100)
training, 60,000 steps	46.4 H100-hours	40.5 A100-hours	2.6 H100-hours
inference memory, 4k context	12.8 MB, any length	12.8 MB	67 MB cache (bf16)

the real gradient, so the matching threshold is $1.0/24$ and the Transformer was clipped twice as tightly. Per-component clipping engages only during loss spikes at these gradient sizes, so we do not expect this to change the comparison. It trains in bf16 mixed precision, the usual setup for this architecture, while the Born model is strict FP32. It keeps the best training-time checkpoint (step 48,000), selected on the same noisy 0.49 MB checks; the Born model’s is its final step, which replaced the best check because a last such check scored at least as well (§3). Its scorer reproduces CNet’s 4,096-byte window protocol exactly: each window is one forward pass, its first byte is unscored, and all models in Table 5 score the same 41,277,600 test bytes.

The Transformer is better by 0.079 bits per byte on test and 0.084 on the validation prefix (Table 5), more than the Born model gains from carrying state over five million bytes (0.012). It also trains 18 times faster per step. That is a comparison of these implementations, bf16 FlashAttention in PyTorch against FP32 CNet kernels, and not of the attention operators. This comparison alone does not identify the cause of the gap: the models differ in complex against real arithmetic, in the attention operator, in the read-out, in precision, and in normalization and initialization. The ablation below separates some of these. The samples show one qualitative difference. Continuing the same 1,500-byte book passage with the same three seeds, the Transformer used only character names that appear in the prompt (18 of 18), while the Born model’s final checkpoint invented new ones (1 of 17 from the prompt), although many of the prompt’s names occur within 1,500 bytes of the continuation. This is one prompt and is evidence of mechanism only in the anecdotal sense; a retrieval test would measure it.

Ablation at 10,000 steps. To separate the operator from the stream, we trained four models in PyTorch with identical bytes, vocabulary, batch, learning-rate schedule (warm-up 500, cosine from 10^{-3} to 10^{-4} over 10,000 steps), optimizer, clipping and seed, and scored them on the same test bytes (Table 6). The real cells are the Transformer above, with softmax or with exact Born attention plus the fixed decay. The complex cells are a PyTorch mirror of CNet’s 12M model (23,933,952 real numbers, CGelu, TokenNorm, Born read-out, rotary positions only), with softmax over $\text{Re}\langle q, k \rangle$ or with CNet’s Born attention plus decay in its exact chunkwise form, trained with CNet’s floor on the read-out ($-\log(p + 10^{-3})$; scoring is exact). A fifth cell gives the complex softmax model a standard linear softmax read-out of the same size. All use bf16 for linear layers and fp32 attention scores. We checked the complex Born cell for leakage: changing byte j leaves every earlier output exactly unchanged, at and around chunk boundaries, and its linear and quadratic forms agree to

Table 6: Ablation at 10,000 steps, bits per byte, 4,096-byte windows, identical byte sets. One seed per cell; all cells see the same batch order.

	test	validation, first 15.7 MB
real stream, softmax	1.495	1.537
real stream, Born + decay	1.511	1.549
complex stream, softmax, Born read-out	1.615	1.656
complex stream, softmax, softmax read-out	1.539	1.576
complex stream, Born + decay, Born read-out (CNet’s design)	1.413	1.457

3×10^{-4} with identical weights.

In this matched five-configuration experiment at 10,000 training steps, the complex-stream model with decayed Born attention and Born read-out achieves the lowest test loss, 0.082 bits per byte below the real softmax baseline. The relative performance of Born and softmax depends on the surrounding representation and read-out: in the real stream Born attention costs 0.017 bits per byte, in the complex stream it gains 0.20, and the Born read-out costs softmax attention 0.077 but works with Born attention. Comparing the 10,000-step mirror directly with the original 60,000-step CNet run of Table 5 (1.324, only 0.09 below the mirror at 10,000 steps, while the Transformer improves from 1.495 to 1.245) confounds training duration with implementation. The next paragraph separates the two, by testing the initialization in CNet’s own code and by training the Gaussian-initialized mirror for the full 60,000 steps. Each ablation cell is a single seed, and a loss spike near step 3,100 that appears in every cell shows that the shared batch order matters.

CNet’s default initialization. The implementation difference that matters is the initialization. CNet’s default sets every weight to $r/n + i\sqrt{1-r^2}/n$, with r uniform in ± 0.5 and n the number of weights in the layer, so each layer starts close to a constant matrix: the largest singular value of a query projection is 50 times its median, against 1.9 for the mirror’s Gaussian start (independent $\mathcal{N}(0, 0.02^2)$ real and imaginary parts). CNet’s block applies CGelu, which halves inputs below about 0.1, twice to the residual stream; with nearly empty residual branches the stream shrinks fourfold per block, to 2×10^{-7} after eight blocks, and the blocks receive gradients four to five orders of magnitude smaller than the read-out. Trained with this start, the PyTorch mirror stays at the byte-frequency plateau for about 2,300 steps. We then trained CNet’s own C++ code on the 10,000-step schedule with both starts, eight A100 GPUs, strict FP32 and everything else as in the 60,000-step run (Table 7). With its default start CNet escapes the plateau quickly but trails its mirror by 0.22 nats per byte at step 2,000; with the Gaussian start it matches the mirror within the noise of the check. The initialization therefore accounts for the difference between CNet and its mirror, and precision (FP32 against bf16) does not. The 60,000-step CNet model of Table 5 was trained with the default start. Training the mirror with the Gaussian start for the same 60,000 steps improves the test score from 1.324 to 1.301 bits per byte, so the initialization accounts for 0.023 of the original 0.079 gap. The Transformer still leads at full length, by 0.056 on test and 0.068 on the validation prefix. The ranking therefore depends on the training budget: with a 10,000-step schedule the complex Born design ends 0.082 ahead, with a 60,000-step schedule 0.056 behind. This is not a statement about early learning speed: on the 60,000-step schedule the Transformer is ahead at every training-time check, for example 1.081 against 1.122 nats per byte at step 10,000. The complex design’s 10,000-step result comes from a schedule that has fully annealed by then. Training-time validation checks during the run put the two within 0.01 near the end; the full test set does not, which shows how little the 0.25–0.5 MB checks can resolve.

Table 7: CNet’s own code with its default initialization and with the mirror’s Gaussian start, against the PyTorch mirror; training-time validation in nats per byte, 10,000-step schedule, same data, batch and seed.

step	500	1,000	1,500	2,000	3,000	3,500
CNet, default initialization	2.233	1.656	1.479	1.405	1.329	1.273
CNet, Gaussian initialization	1.494	1.268	1.218	1.180	1.143	1.110
PyTorch mirror, Gaussian initialization	—	1.297	—	1.184	1.136	—

The Born model keeps two advantages. Its state is a fixed 12.8 MB for any input length, and each new byte costs the same, whereas a Transformer needs its key/value cache, 67 MB in bf16 for a 4,096-byte window, and attention over the whole window per byte. Neither is a quality advantage. A sliding-window Transformer also runs in constant memory over an unbounded stream, and Section 6 shows that the Born model uses no more context than such a window.

6 Reset sweep

Table 8 reports $\Delta L(C)$ for both models on identical validation bytes. Carried state helps both, and the 12M model about twice as much: -0.0123 against -0.0064 bits per byte at $C = 5\text{M}$. The sign and size agree across the three 5M-byte windows (per-window values within 0.0007 of the overall figure); the windows are three stretches of one file, not independent replications. Deltas are comparable within each model; the models’ absolute losses differ also through their vocabularies (174 and 222 classes).

The shape of the curves is informative. Suppose a reset costs a fixed number of bits c , spent while the state is rebuilt after the boundary, and that nothing older than the previous few thousand bytes is used. A file of n bytes reset every C bytes then pays cn/C , so

$$\Delta L(C) = \Delta L(\infty) \left(1 - \frac{4096}{C}\right), \quad (2)$$

which is 50%, 75%, 87.5% and 94% of the full gain at 8k, 16k, 32k and 64k. With $\Delta L(\infty)$ taken from the 5M point the model has no free parameter left, and both measured curves follow it with a largest residual of 3×10^{-5} (4M) and 4×10^{-5} (12M) bits per byte (Table 8, “predicted” rows), with $c = 26$ bits per reset for the 4M model and $c = 50$ for the 12M model. The data are therefore consistent with most of the aggregate benefit being recovery after a reset: the first few hundred to few thousand bytes after a boundary predicted without the preceding text. Extending the reset interval from 256k to 5M bytes improves the 12M model’s average loss by about 1.8×10^{-4} bits per byte, closely matching the fixed-cost model’s prediction. The residual is below 4×10^{-5} bits per byte. This agreement supports the fixed-cost-per-reset description, but does not isolate the contribution of long-range information: such information could be part of the measured improvement and be absorbed into the fitted cost. The 12M model pays more per reset; its slowest head has a half-life of 709 bytes against 177 for the 4M model, so its state takes longer to refill, though the larger model also uses its state differently and the sweep cannot separate the two.

A curve that fits (2) does not identify the mechanism. It does not show that every head has forgotten everything beyond a few kilobytes, and long-range information that matters for a few predictions could leave the average almost unchanged; a retrieval test is needed for that (§10). Resetting at positions shifted against natural text boundaries would also show whether the cost depends on where a stream starts.

Table 8: Reset sweep: $\Delta L(C) = L(C) - L(4,096)$ in bits per byte on the first 15,728,640 validation bytes. Negative means carried state helps. $L(4,096)$ on this 15.7 MB prefix is 1.7065 (4M) and 1.3712 (12M); the whole-split 4M figure in Table 4 (1.696) covers different bytes. Predicted rows use only the measured 5M value.

reset interval C	8k	16k	32k	64k	256k	5M
4M model, measured	-0.0032	-0.0048	-0.0056	-0.0060	-0.0063	-0.0064
4M model, predicted by (2)	-0.0032	-0.0048	-0.0056	-0.0060	-0.0063	—
12M model, measured	-0.0062	-0.0092	-0.0108	-0.0115	-0.0121	-0.0123
12M model, predicted by (2)	-0.0061	-0.0092	-0.0107	-0.0115	-0.0121	—

7 Decay along dependency paths

Within one layer. By (1), a term written into a head’s state at position s is multiplied by γ_h^{t-s} by position t . Nothing in the layer refreshes it, and a summary formed by the layer is itself a term written at some position and decays from there. The direct contribution of an old input is therefore suppressed exponentially. Two caveats keep this from being a statement about information. The output is a ratio, and a decayed old key can still dominate if every recent key scores near zero against the query. And a small term need not be useless if later layers can decode it.

Across layers. Between state updates, everything in a block acts on one position at a time: the normalization, the feed-forward layer, the projections and the residual connection.

Proposition 1. *In a stack of L such blocks, every dependency path from the input at position s to the output at position $t > s$ advances in position only inside state updates, at most once per layer, so it crosses at most L states. If the path crosses $D_\ell \geq 0$ positions in layer ℓ ($D_\ell = 0$ where it takes the residual connection or stays at one position), then $\sum_\ell D_\ell = t - s$, and the product of the decay factors along the path is at most $\gamma_{\max}^{t-s}$, where $\gamma_{\max}$ is the slowest decay in the model.*

Proof. Let x_t^ℓ be the input of block ℓ at position t , with x_t^0 the embedded byte. Block ℓ computes $x_t^{\ell+1} = f_\ell(x_t^\ell, y_t^\ell)$, where f_ℓ (normalization, projections, residual connection, feed-forward layer) acts on position t alone, and by (1) the attention output y_t^ℓ depends on $\{x_{s'}^\ell : s' \leq t\}$ only through terms written at s' and scaled by $\gamma_h^{t-s'}$. Draw a layered graph with nodes (ℓ, t) and an edge $(\ell, s') \rightarrow (\ell + 1, t)$ for every $s' \leq t$, labelled $\gamma_h^{t-s'}$ for the state route and 1 for the position-wise route ($s' = t$). Every edge goes from layer ℓ to layer $\ell + 1$, so a path from $(0, s)$ to (L, t) has exactly L edges, one per block, with advances $D_\ell = t_{\ell+1} - t_\ell \geq 0$ that sum to $t - s$. The product of its labels is $\prod_\ell \gamma_{h_\ell}^{D_\ell} \leq \gamma_{\max}^{\sum_\ell D_\ell} = \gamma_{\max}^{t-s}$. $\square$

The proposition bounds the explicit decay factor along a dependency path, not the influence of an input on the prediction. The network is not a linear sum over paths: the attention output is a ratio, the position-wise maps are nonlinear and can amplify a weak signal, and a later layer can write an amplified copy into its own state, where it decays afresh. Relaying through the stack cannot escape the decay in the product of factors, only through such amplification.

An illustrative estimate. Amplification cannot recover a signal that the state no longer represents. The state is FP32 and also holds the recent bytes, about $1/(1 - \gamma)$ terms of similar size. A decayed term below roughly 2^{-24} of the state’s magnitude is lost to rounding, which for the slowest head of the 12M model happens between about 10k bytes (counting the recent mass) and 17k bytes

Table 9: Claims and the evidence for each.

claim	status	evidence
Exact streaming with fixed-size state	structural	recurrence (1); 12.8 MB state for the 12M model
5M-byte inference without degradation	demonstrated	Table 4, 5M windows vs 4k resets
5M-byte effective memory	not supported	reset sweep (Table 8): 1.8×10^{-4} bits/byte from 256k to 5M, consistent with a fixed reset cost, long-range contribution not isolated; decay (Table 1) limited by the decay design; see §10
Long-range retrieval	not tested	better at 10k (1.413 vs 1.495); worse at 60k with a Gaussian start (1.301 vs 1.245, Table 5)
Quality equal to a matched Transformer	not supported at full length	

(ignoring it). With $L = 8$ hops a naive reading gives roughly $8 \times 16k \approx 130k$ bytes. This is an illustration of scale, not a bound on information flow. Rounding depends on the magnitude of each component, the accumulation order and cancellation, and the complex off-diagonal entries of the state can cancel; a formal bound would also have to model normalization and the nonlinearities. The measured sweep, not this estimate, is the evidence for a short practical horizon.

8 What the evidence supports

Table 9 sorts the claims by kind. Constant-size state and stable streaming are properties of the method and the implementation, measured on this hardware. Loss values, the reset sweep and comparisons between mixers are modelling evidence. They support different statements and are reported separately.

9 Limitations

- The matched baseline differs from the Born model in precision (bf16 against FP32) and in checkpoint selection (best training-time check at step 48,000 against the final step, kept by a last noisy check). Selecting on 0.49 MB checks favours the Transformer slightly; the gap of 0.079 bits per byte is far larger than that noise. A complex gated linear attention model of the same size was not trained; the mixer comparisons in the companion paper [1] are at the 4M scale.
- Each configuration was trained once, with one seed. The ablation cells share one batch order, so they are paired but not independent; seed variance was not measured.
- The well-initialized 60,000-step model is the PyTorch mirror (bf16 linear layers, its best training-time checkpoint at step 58,000); CNet itself has not been retrained for 60,000 steps with the Gaussian start.
- The reset sweep uses one contiguous validation prefix of 15.7 MB. The three per-window deltas are a small repeatability check, not a sample of all validation text. The test set is used only for the headline score.
- The aggregate loss can hide long-range ability or its absence. No retrieval test was run.

- Step-time statistics are 100-step means; data-loader wait, peak memory beyond the static arena and scaling efficiency were not measured.

10 Future work

Changing the decay. The decay sets a trade-off between forgetting, which cured attention dilution in the companion paper, and retention. The direct experiment keeps the model and protocol fixed and varies only the slowest heads: one or two heads with γ close to 1 (for example $1 - 2^{-16}$, a half-life of about 45k bytes), or learned decay rates, compared with the fixed schedule on validation loss and on the reset sweep.

Measuring influence directly. A perturbation probe changes one byte at distance D before position t and measures the change in the predicted distribution at t , averaged over positions and replacement bytes, per layer and head. It needs only forward passes. A nonzero effect shows influence, not useful retrieval; a zero effect needs a noise floor measured with $D = 0$ controls and repeated FP32 runs.

Retrieval at controlled distances. A passkey-style test [5] places a short string at distance $D \in \{4k, 16k, 64k, 256k\}$ before a query, with distractors and random positions. On the current models the result is predictable from Table 1; the test becomes informative for the slow-decay variants above.

Closing the gap at full length. The design wins with a short schedule and loses with a long one; whether that holds for other model sizes is open, and each configuration was trained once. Independent seeds with full-split scoring, compute-matched comparisons (accelerator-hours, not only steps), and validation on contiguous, book-stratified blocks instead of small random chunks would make the ranking robust. Within the complex stream, the remaining components (CGelu with TokenNorm, the read-out and the positional setup) can each be swapped into the real Transformer, and the most informative cells repeated over several seeds.

11 Conclusion

A constant-state Born-attention byte model reads five-million-byte windows in one pass with 12.8 MB of state, and on PG-19 its loss does not degrade over the window. The fixed decay that makes this model train well also suppresses old contributions exponentially, and the reset sweep finds little aggregate benefit from persistence beyond a few thousand bytes: the gain follows a fixed cost per reset closely, and extending persistence from 256k to 5M bytes improves the average loss by only about 1.8×10^{-4} bits per byte, though this does not isolate the contribution of long-range information. After 10,000 steps the complex Born design beats a matched Transformer by 0.08 bits per byte; after 60,000, with a properly initialized model, the Transformer leads by 0.06. Streaming, memory and quality are separate properties. This method delivers the first; the second depends on the decay; the third it reaches with a short training budget but not with a long one, at this size.

A Reproducibility

Code: <https://github.com/crasmarum/CNet>, built from commit d0955d1f of the development repository for the 12M run (binary built on the node; the sources of the trainer, the Born kernels and

the Makefile match that commit by SHA-256). The 12M checkpoint is 95,741,704 bytes: 11,966,976 complex parameters at 8 bytes each plus a 5,896-byte graph header, with no optimizer state. SHA-256: 5caee072a6fbe13b3b962ea61ba4d1226c0d9deaa1851d6373db4280694cba34. Hardware: one node of 4× NVIDIA H100 SXM5 80 GB HBM3 with NVLink, two Xeon Platinum 8480+ CPUs, driver 580.105.08, CUDA 12.8.

Training (12M; launched with `run_dp.sh 4`):

```
dp_shakespeare -data pg19_train.txt -val_data pg19_val.txt -emb 384 -heads 6
  -tokens 4096 -blocks 8 -batch 24 -steps 60000 -warmup 2000 -lr 1e-3
  -min_lr 1e-4 -rope true -pos_table false -token_norm true -tw_cublas true
  -fft_fast true -born_chunk 512 -ce_eps 0.001 -grad_clip 0.5 -seed 1000
  -val_every 2000 -born true -born_linear true -born_decay true
```

Carried-state window score (reset sweep: `-eval_window C -eval_bytes 15728640` on the validation file):

```
dp_shakespeare -tw_cublas true -fft_fast true -born_chunk 512
  -eval_stream pg19_test.txt -model pg19_full_born.mod
  -eval_window 5242880 -eval_batch 8
```

Transformer baseline (`scripts/pg19_full/transformer_baseline.py`; scoring with `--eval FILE`):

```
torchrun --nproc_per_node 4 transformer_baseline.py --train pg19_train.txt
  --val pg19_val.txt --vocab pg19_full_born.mod.vocab --save tf_baseline.pt
```

The same script runs the ablation cells and the mirror: `--attn born` swaps in exact Born attention with the fixed decay; `--arch complex` uses the PyTorch mirror of CNet’s model (`cnet_torch.py`), with `--born_linear` for the chunkwise Born form, `--readout softmax` for the read-out swap, `--ce_eps 0.001` for CNet’s training floor and `--cinit cnet` for CNet’s old initialization. The ablation cells use `--steps 10000 --warmup 500`; the 60,000-step mirror uses the CNet run’s schedule (`--steps 60000 --warmup 2000`) on eight GPUs. CNet’s Gaussian initialization is `dp_shakespeare -init_std 0.02`, now the default (`-init_std 0` restores the old start). Run scripts and logs are in the `results/` folders of the development repository.

B Samples

Exploratory checkpoint comparison. We sampled the checkpoints of steps 8k, 26k and 60k with the same two prompts and the same three seeds and scored each continuation with GPT-2 (124M) conditioned on its prompt. Perplexity falls from 464 to 311 to 162, against 134 for the real continuation of the book prompt, while the share of generated words that occur in the validation text stays at about 98%: spelling is learned early, and later gains are in phrase and sentence structure. GPT-2 perplexity is not a measure of coherence, low values can reward bland or repetitive text, and the reference is a single passage. The Transformer baseline’s samples score 98, partly through repetition.

Step-26,000 checkpoint of the 12M model (validation 1.033), CPU constant-state decoder, temperature 0.8, continuing a passage from the test partition (the bracketed letter ends the prompt):

“[H]ave you not gone to much rest, or save her?”

“Yes, dear Mrs. Trion,” said the Countess; “so I fancy you do not care to talk to me on that sort

of thing, but two shall not consider you the less uncomfortable.”
 “She forgives you, sir, wherein I will promise you to take the same book with my own.”
 ...
 “Thank you, Jeannette, I cannot believe,” said Mitrieux, moaning for some time. “Heaven may never let you tell me, gentlemen, that you are not the only consignment that is to be felt.”

Final checkpoint (step 60,000), same prompt and temperature:

“[H]a! Faith, and am I to reign a god of my grandeur, and to like your gentle Carlos and come to the age of seven! Get me the Shan into your confidence, for the gentle dog that crusheth her royal godmother’s daughter as mother in the days of her mammon: that I will be as noble as on her friends! But if the day comes to my left tower for the first time, you must excuse it, and so from the day I may venture to have now been a play on the part of the Princess of Worcester, who will have no love for her; ...”
 “See her!” said the vicar, “I will take her, and be ready, I say with all my soul; ...”

Matched-seed samples from steps 8k, 26k and 60k, with the GPT-2 scorer, are in the results folder of the run.

References

- [1] Marcel Crasmaru. Born attention is linear: Exact sub-quadratic attention and constant-state inference from quantum measurement, 2026. Preprint.
- [2] Marcel Crasmaru. CNet: A complex-valued deep learning framework with wirtinger autodifferentiation and FFT–hadamard convolution. *arXiv preprint arXiv:2610.08592*, 2026. URL <https://arxiv.org/abs/2610.08592>.
- [3] Eric Egli, Matteo Manica, and Jannis Born. Multiscale byte language models: A hierarchical architecture for causal million-length sequence modeling. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2502.14553.
- [4] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning (ICML)*, 2020.
- [5] Amirkeivan Mohtashami and Martin Jaggi. Landmark attention: Random-access infinite context length for transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.
- [6] Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations (ICLR)*, 2020. Introduces the PG-19 benchmark.
- [7] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. 2024.
- [8] 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.

- [9] Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *International Conference on Machine Learning (ICML)*, 2024. arXiv:2312.06635.
- [10] Lili Yu, Dániel Simig, Colin Flaherty, Armen Aghajanyan, Luke Zettlemoyer, and Mike Lewis. MEGABYTE: Predicting million-byte sequences with multiscale transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.