

Four Kilobytes of Memory: A 10M-Parameter Fixed-State Recurrent LM for BabyLM Strict-Small

Sam Furr

samneillfurr@gmail.com

Abstract

How much language competence can a small recurrent model acquire when it must compress its context into a fixed-size state? We study this question with StateHead, a gated recurrent language model without attention or positional embeddings. Our BabyLM 2026 Strict-Small submission has 9.97M parameters and 2,160 recurrent state values, equivalent to 4.2 KiB in bf16. After 1.1 hours of training on a laptop, it scores 68.2 on BLiMP, compared with 65.2 for the official 98.4M-parameter GPT-2 baseline on the same benchmark suite. This advantage is specific to grammatical discrimination: StateHead is weaker on reading-time prediction and scores near or below chance on several semantic and entity-tracking evaluations. We characterize its computation and training trajectory, and compare recipes that reset or carry state across training chunks. The apparent entity-tracking gain is confined to items whose correct answer is “nothing”; both recipes score below chance on the revised benchmark. These results document both the useful capabilities and the limits of a language model with a very small recurrent memory.

1 Introduction

The BabyLM Challenge asks how much language competence can be learned from a limited amount of experience (Warstadt et al., 2023; Hu et al., 2024; Charpentier et al., 2025). Its 2026 Strict-Small track allows a corpus of at most 10M words and at most 100M word exposures during model training (Choshen et al., 2026). This setting makes it possible to study architectural choices with modest resources, while testing grammatical, semantic, and processing-related generalization.

We examine one such choice: compressing preceding context into a recurrent state whose size does not grow with the sequence. A transformer with full attention can retain a token-indexed record of its context. A recurrent model instead updates a

fixed set of values as each token arrives. This constraint reduces persistent inference memory, but also requires the model to retain useful distinctions without direct access to earlier token representations.

Incremental comprehension provides one motivation for studying this constraint. The Now-or-Never account emphasizes that transient linguistic input must be encoded as it arrives (Christiansen and Chater, 2016), and recurrent networks have long been used to study learning from sequential input (Elman, 1990). The analogy has limits. Human readers can revisit a passage, and a fixed-state text model does not reproduce human memory or eye movements. We treat bounded state as a computational restriction to investigate, rather than evidence of cognitive fidelity. Reading-time prediction is an empirical test of alignment, and our model performs poorly on that test.

StateHead combines a gated recurrent update with a residual language-model backbone. Its recurrence admits parallel scanning during training and carries a vector of state values per layer. We study a 10M-parameter submission and a 98M-parameter probe. Recurrent syntax learning has prior support (Gulordava et al., 2018); our question is how far this particularly small state can support the broader BabyLM suite.

We document the capabilities and limitations of this system, quantify its computation and persistent memory, and report its training trajectory. We also trace an apparent benefit from a state-carry recipe to entity-tracking items removed by an evaluator revision. Our contribution is empirical: it characterizes what this restricted model learns, rather than proposing a new recurrent-cell principle.

	GPT-2	SH-10M	SH-98M
Parameters	98.4M	9.97M	98.7M
Forward FLOPs/token	233.6M	20.0M	197.5M
Attention/scan share	16.4%	0.22%	0.09%
Training FLOPs	—	9.7×10^{15}	9.7×10^{16}
State at 1,024 (bf16)	36.0 MiB	4.2 KiB	17.4 KiB
State at 4,096 (bf16)	144.0 MiB	4.2 KiB	17.4 KiB
Training time	—	1.1 h	5.1 h
Training tokens/s	—	39.8k	9.0k

Table 1: Analytic costs and measured StateHead training times. State sizes assume two bytes per value and exclude parameters, training activations, and workspace. The 4,096-token GPT-2 cache is a size extrapolation beyond its 1,024 learned positions. Training FLOPs use three times forward cost. GPT-2 training time was not measured.

2 StateHead

2.1 Architecture and recurrent update

StateHead uses pre-normalized residual blocks. Each block contains a recurrent token mixer followed by a feed-forward multilayer perceptron (MLP), with a residual connection around each sublayer. The input and output embeddings are tied (Press and Wolf, 2017). There are no positional embeddings: order enters through the recurrent update. Figure 1 shows the block and its gates.

Let d denote the residual width, H the number of heads, and $d_h = d/H$ the width of each head. For a normalized token representation $x_t \in \mathbb{R}^d$, one affine projection produces all four sets of pre-activations:

$$\begin{bmatrix} u_{a,t} \\ u_{b,t} \\ u_{c,t} \\ u_{o,t} \end{bmatrix} = W_g x_t + \beta_g, \quad W_g = \begin{bmatrix} W_a \\ W_b \\ W_c \\ W_o \end{bmatrix}. \quad (1)$$

Here $W_g \in \mathbb{R}^{4d \times d}$ vertically stacks four $d \times d$ matrices; it is not an additional projection before those matrices. The learned bias β_g is partitioned in the same way. Each output is then split into H heads. Each head maintains $s_t \in \mathbb{R}^{d_h}$ and produces $y_t \in \mathbb{R}^{d_h}$. With σ denoting the logistic sigmoid and $\odot$ elementwise multiplication, and suppressing the head index, the update is

$$\begin{aligned} a_t &= \sigma(u_{a,t}), & b_t &= \sigma(u_{b,t}), \\ c_t &= \tanh(u_{c,t}), & o_t &= \sigma(u_{o,t}), \\ s_t &= a_t \odot s_{t-1} + b_t \odot c_t, & y_t &= o_t \odot s_t. \end{aligned} \quad (2)$$

The retention gate a_t scales the previous state, while $b_t \odot c_t$ adds new information. The out-

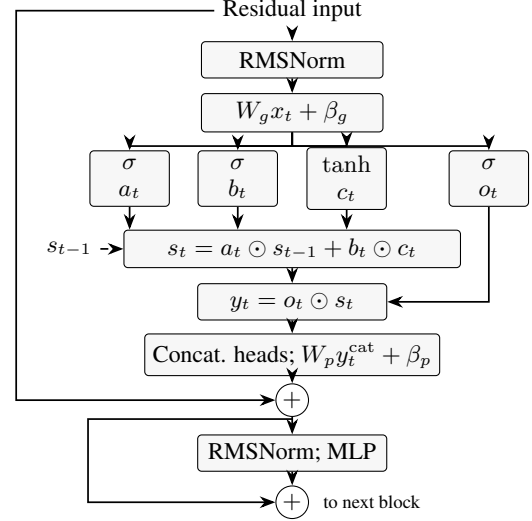


Figure 1: One StateHead block. The fused projection splits into retention, write, candidate, and output branches. Each head updates a vector state; y_t^{cat} concatenates the gated head outputs. The dashed arrow supplies the previous state, and s_t is retained for the next token. The outer paths are residual connections. Stacking L blocks requires Ld recurrent values.

put gate controls which state coordinates are exposed to the rest of the block. All state operations are elementwise. Head outputs are concatenated into $y_t^{\text{cat}} \in \mathbb{R}^d$ and mixed by the affine projection $W_p y_t^{\text{cat}} + \beta_p$ before the residual addition, with $W_p \in \mathbb{R}^{d \times d}$ and learned bias $\beta_p \in \mathbb{R}^d$.

Retention biases start at 2, giving an initial sigmoid value of about 0.88 at zero input; the other gate biases start at zero. Each layer has a learned initial state used at resets. Each head’s gates use all d input coordinates despite its diagonal state transition. At fixed width, parameters, and flattened state, the head partition leaves the mathematical forward computation unchanged; it determines the gate-matrix grouping for head-wise Muon (Section 4.2).

Once the gates are computed, the update is affine in s_{t-1} . Affine updates compose associatively, permitting a parallel prefix scan (Blleloch, 1990; Martin and Cundy, 2018). The implementation uses scan chunks of 64 tokens and fp32 accumulation. This scan chunk size is an implementation parameter, distinct from the 1,024-token training windows described below.

2.2 Computation and persistent memory

Let N be the number of unique trainable parameters, L the number of layers, and T the context length. We count a multiply-accumulate as two

floating-point operations (FLOPs). Table 1 summarizes the accounting; Appendix B gives formulas and a profiler cross-check.

Most computation is in the gate projections, output projections, MLPs, and vocabulary projection. Under serial-equivalent arithmetic accounting, the elementwise recurrence and gate activations contribute approximately 0.22% of forward FLOPs for SH-10M and 0.09% for SH-98M. Our accounting gives a total per-token cost close to $2N$. This follows the dense-matrix convention of Kaplan et al. (2020), but our N includes the tied embedding/output matrix and our cost includes vocabulary logits. These percentages describe the mathematical update, not the exact instructions of a parallel kernel, which can repeat operations. Scan dependencies and memory traffic also affect runtime, so operation share should not be read as runtime share.

StateHead carries Ld values independently of T . SH-10M therefore needs 2,160 values, or 4,320 bytes (4.2 KiB) in bf16. This is persistent sequence state, not total model memory: weights, activations, and temporary buffers are additional. The scan accumulates internally in fp32 and returns state in the numerical precision used for execution. Running in fp32 uses 8.4 KiB for the same state; the exported configuration defaults to fp32.

Full-attention GPT-2 retains $2LTd$ key and value entries. The baseline uses 36 MiB of bf16 cache at $T = 1,024$, and attention adds context-dependent computation. The large FLOP difference between SH-10M and GPT-2 is primarily due to parameter count; removing full attention accounts for a smaller additional saving. Fixed-window attention can also bound memory as a function of total sequence length. StateHead’s more restrictive property is that it stores no token-indexed cache at all.

3 Related Work

Gated recurrent models. QRNNs combine temporal convolutions with elementwise recurrent pooling (Bradbury et al., 2017). StateHead’s cell uses the same update form as QRNN *ifo* pooling: independent retention and write gates, a bounded candidate, and an output gate. Here the projections use only the current token, equivalent to width-one convolutions, within a pre-normalized residual/MLP backbone. RG-LRU also uses a diagonal transition, but parameterizes retention through a learned decay and scales writes by $\sqrt{1 - a_t^2}$ (De

et al., 2024). StateHead does not use that write scaling. Its state is a vector of d scalars per layer; matrix-memory models such as mLSTM store associations through outer-product updates (Beck et al., 2024).

Recurrent models in BabyLM. BLaLM is a close predecessor (Haller et al., 2025). It replaces self-attention with mLSTM and studies short convolutions, sliding-window attention, and other additions. Its 270M-parameter Strict-Small submission reports BLiMP 67.0, EWoK 50.6, and COMPS 50.5. This profile resembles our stronger grammatical discrimination and weak performance on broader semantic tests. The numbers cannot establish a ranking: BLaLM uses a curated 2025 corpus, a larger model, and an earlier evaluator. Its entity-tracking score of 33.7 predates the filtering change in Section 5.3.

Haller et al. also provide controlled evidence about optimization. In their Strict-track experiment, Muon reduces perplexity from 11.21 ± 0.11 to 7.95 ± 0.15 across three runs, while mean zero-shot performance changes more modestly, from 35.75 ± 1.74 to 36.24 ± 1.16 . Our use of Muon follows that precedent. We do not establish an additional benefit from our head-wise update rule, which was not ablated.

Context length and recall. Salhan et al. (2025) study OPT and Mamba on the 100M-word Strict corpus and find task-dependent, non-monotonic effects of training length. Their Table 2 distinguishes a training-optimal length, balancing cost and performance, from the length with the highest score. For Mamba entity tracking these are 1,024 and 128, respectively; reading prediction peaks at 64. For OPT, the highest entity score instead occurs at 8,192. Their results therefore do not support a universal preference for long recurrent training contexts. Our 1,024-token setting is within a previously useful range, but its suitability must be checked for StateHead and each task.

Lee et al. (2025) find complementary recall behavior in Mamba–attention hybrids. Sequential hybrids tend to favor shorter-context recall, whereas parallel hybrids with attention-based merging perform better at longer contexts. Those experiments use 430M–1.3B parameters and 100B SlimPajama training tokens, providing architectural evidence rather than a Strict-Small baseline. Their findings motivate studying explicit retrieval. StateHead examines the more restrictive setting without any lo-

cal or global attention cache.

4 Experimental Setup

4.1 Data, model sizes, and training windows

Both models train for ten epochs on the BabyLM 2026 Strict-Small corpus, nominally 100M word exposures. Byte-pair tokenizers (Sennrich et al., 2016) are trained only on that corpus. SH-10M uses an 8,192-token vocabulary; SH-98M uses 16,384 tokens. Tokenizer training does not count toward the model-training exposure allowance (organizer Mustafa Omer Gul, personal communication, July 9, 2026). Training logs record 161.2M and 163.2M token exposures, respectively.

Table 2 gives the configurations. The larger probe approximates the parameter count of the official GPT-2 baseline. The smaller configuration targets an approximately 10M-parameter system and was selected after development experiments with alternative widths, depths, vocabulary sizes, and MLP ratios. It was not chosen through a controlled scaling experiment. Because parameter allocation and tokenization both change, the models test two practical configurations rather than isolate parameter count alone.

The 1,024-token window controls packing, the number of tokens processed together, and the distance through which gradients propagate. Linear-time recurrence does not remove training activation costs or the need to choose a backpropagation window. Inference state can continue beyond that length, but this does not demonstrate accurate recall at arbitrary distances. Development runs also examined windows of 512 and 2,048 tokens; Appendix C describes their narrower scope. Dropout was fixed at zero and was not ablated. We make no claim that either setting is optimal for this data budget.

4.2 Training recipe and checkpoint selection

Training uses MLX (Hannun et al., 2023) on an Apple M5 MacBook Air with a 10-core GPU and 32 GB of unified memory. The microbatch is one window, with 32 accumulation steps, yielding 32,768 tokens per full optimizer update. Parameters and activations use bf16 where supported, and the scan accumulates in fp32. Reported times measure the training loop and exclude tokenizer preparation, initial data loading, checkpoint export, and evaluation.

Setting	SH-10M	SH-98M
Layers L	6	12
Heads H	6	12
Residual width d	360	744
Head width d_h	60	62
MLP expansion r	2	4
Vocabulary V	8,192	16,384
Unique parameters N	9,971,640	98,658,120
Training window T	1,024	1,024
Dropout	0	0
State between windows	Reset	Carry

Table 2: Model configurations. The GELU MLP maps $d \rightarrow rd \rightarrow d$, so r is the hidden expansion ratio. Both models use RMSNorm, tied embeddings, and learned initial states, without positional embeddings. State carry is detached and resets at document boundaries.

SH-10M concatenates corpus rows without separators and divides the stream into windows. It resets to the learned initial state at every window, including windows that cross a row boundary. The loss therefore covers transitions between rows. SH-98M carries detached state across consecutive chunks within documents and resets at document boundaries. Both shift the packing offset between epochs to vary window boundaries. These differences also confound comparison of the two scales.

Hidden weight matrices and $H \times d_h$ learned initial-state matrices use Muon (Jordan et al., 2024). Token embeddings, tied output weights, normalization scales, and biases use AdamW (Loshchilov and Hutter, 2019). For W_g , the implementation groups update rows by head and applies Muon to H matrices of shape $4d_h \times d$. Each group contains that head’s rows from all four gates. Ordinary Muon would process the complete $4d \times d$ matrix. Unit tests verify this grouping against explicit slices, but we did not compare the variants experimentally. Head-wise Muon is a disclosed recipe choice rather than a demonstrated improvement.

We save 19 checkpoints at nominal word-exposure milestones: every 1M through 10M, then every 10M through 100M. Labels follow optimizer-update progress, not a whitespace-word counter. SH-10M is submitted at its final checkpoint. The main SH-98M result uses a weight average of its 60M, 80M, and 100M milestones (Izmailov et al., 2018), selected before full evaluation.

4.3 Evaluation and comparison scope

We use the official 2026 benchmark suite at its July revision (BabyLM Organizing Committee, 2026). Zero-shot tasks include BLiMP (Warstadt

Setting	Value
Schedule	Cosine; 200 warmup steps
AdamW learning rate	5×10^{-4} to 5×10^{-5}
Muon learning rate	2×10^{-2} to 2×10^{-3}
Weight decay	0.1
Muon momentum	0.95, Nesterov
Newton–Schulz iterations	5
Updates (10M / 98M)	4,920 / 4,990

Table 3: Optimization settings for the two main runs. Partial accumulation windows at epoch boundaries account for the small difference between update counts and token exposures divided by 32,768.

et al., 2020), its Supplement, EWoK (Ivanova et al., 2025), entity tracking (Kim and Schuster, 2023), COMPS (Misra et al., 2023), and Global-PIQA (Chang et al., 2025). The suite also tests reading-time prediction (de Varda et al., 2024), word-level age of acquisition (Chang and Bergen, 2022), and fine-tuned classification on (Super)GLUE (Wang et al., 2019b,a).

Official dataloaders and scorers drive our MLX zero-shot evaluator. Its scan operations are parity-tested against the PyTorch export. GLUE uses an MLX reimplement with official task defaults; a check against the official RTE path gives similar results. AoA and the GPT-2 baseline use the official PyTorch evaluator. These checks support the comparison but do not independently reproduce every task in both runtimes.

We evaluate the released GPT-2 checkpoint on the same benchmark suite. Our BLiMP score of 65.2 agrees with the published 65.23. The baseline was not retrained with our tokenizer or optimizer, so the comparison concerns complete systems rather than isolating recurrence. All training runs use one seed. Development used the pipeline’s fixed reduced (“fast”) subsets; final results use full sets. Fast and full scores are labeled throughout.

5 Results

5.1 Grammatical discrimination and broader evaluation

Table 4 reports full-set evaluation. SH-10M scores 68.2 on BLiMP, 3.0 points above the official GPT-2 checkpoint with approximately one tenth the parameters. The SH-98M weight average scores 69.9. These are descriptive single-run comparisons, without an estimate of variation across training seeds.

The advantage is concentrated in BLiMP. GPT-2 scores higher on the Supplement, COMPS, and en-

Benchmark	GPT-2	SH-10M	SH-98M
Parameters	98.4M	9.97M	98.7M
BLiMP	65.2	68.2	69.9
BLiMP Supplement	57.3	55.3	56.9
EWoK	50.6	50.6	50.1
Entity tracking	19.1	18.8	18.4
COMPS	51.8	50.8	50.9
Reading (ET / SPR)	9.6 / 1.6	0.2 / 0.1	—
Global-PIQA parallel	—	26.2	—
Global-PIQA nonparallel	—	45.0	—
AoA	—	−0.17	—
GLUE (mean accuracy)	—	56.6	—

Table 4: Full evaluation on the July 2026 benchmark suite. SH-10M is the submitted final checkpoint; SH-98M is the 60M/80M/100M weight average. Dashes indicate results not reported under this protocol. Entity tracking uses the revised full set (five choices; chance 20). Global-PIQA chance is 25 for parallel and 50 for nonparallel. Reading scores are normalized incremental $R^2 \times 100$; AoA is acquisition-curve fitness. GLUE uses the official scorer’s mean per-task accuracy.

tity tracking, although the latter two remain close to chance for all three systems. StateHead’s EWoK and Global-PIQA scores also remain close to or below their nominal chance levels. The results support grammatical discrimination under a small parameter budget, but do not establish broad language understanding.

Reading-time prediction gives the clearest contrasting result. GPT-2 reaches 9.6 on eye tracking (ET) and 1.6 on self-paced reading (SPR), while SH-10M reaches 0.2 and 0.1. Recurrence and competent BLiMP performance are therefore insufficient to produce surprisal estimates that predict the measured reading times well in this setup. We have not identified whether architecture, training, tokenization, or their interaction explains this gap.

Fine-tuned classification is also weak. SH-10M obtains 56.6 mean accuracy and exhibits majority-class behavior on several tasks, including near-zero Matthews correlation on BoolQ, QQP, and MultiRC. Its causal objective and single-token pooling are possible contributors, but neither was ablated. Hybrid masked/causal objectives offer a relevant precedent (Charpentier and Samuel, 2024); our results do not determine which change would improve classification.

5.2 Changes across training

Figure 2 shows BLiMP on the fast subsets over 19 milestones. SH-98M rises to 72.6 near four epochs, then declines to 67.0 at the final individual checkpoint. SH-10M reaches 69.0 and finishes at 67.8,

remaining between 67.8 and 69.0 from the nominal 40M milestone onward. Full-set scores differ from fast scores because they evaluate different samples.

This contrast is consistent with overtraining under repeated exposure, but it does not identify a capacity effect: tokenizer, depth, width, MLP ratio, and state handling all differ. A final-checkpoint comparison can therefore conceal substantial differences in trajectory. The smaller model offered a practical submission with good final performance and low training cost. We do not infer that small models are generally more stable or that the larger model’s best fast-subset checkpoint would be best on the full suite.

Original entity-tracking scores vary sharply, including jumps from 21.6 to 17.2 to 41.4 over adjacent early SH-98M milestones. These scores predate the filtering change discussed next. We retain the milestone record in Appendix D, but do not interpret its old entity scores as evidence of changes in tracking ability.

5.3 Reassessing a state-carry recipe

State handling determines what can persist across training windows. Resetting at each window discards earlier state; carrying detached state within a document preserves forward information without backpropagating through the preceding chunk. This resembles cached segments in Transformer-XL (Dai et al., 2019) and truncated backpropagation through time (Williams and Peng, 1990).

We compared two earlier 98M training recipes, distinct from the averaged SH-98M model reported in Table 4. Both used cosine decay, but the carry recipe also introduced learned initial states and epoch packing shifts. The reset recipe used a fixed zero state and did not use those shifts. This comparison therefore does not isolate state carry. The recipes gave similar fast-subset language scores: reset/carry results were 66.6/66.2 on BLiMP, 56.0/55.2 on the Supplement, and 48.6/49.0 on EWoK. On the original fast entity benchmark, however, the macro-average rose from 17.0 to 42.0. Taken alone, that difference would suggest a benefit from the combined recipe.

The July 8, 2026 evaluator update (revision 7d0a6c8) excludes every item for which any candidate answer contains “nothing” (BabyLM Organizing Committee, 2026). In the original fast *regular* split, it removes 914 of 3,152 items (29.0%). We partitioned saved predictions by that exact criterion. Table 5 shows that the gain is confined to the re-

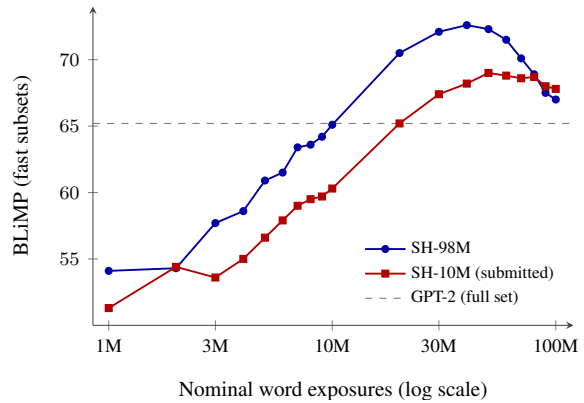


Figure 2: BLiMP on fixed fast subsets across nominal word-exposure milestones. The dashed reference is GPT-2’s full-set score, shown for orientation because the samples differ. SH-98M points are individual checkpoints, not the weight average in Table 4.

Original fast-split group	Items	Reset	Carry
Retained by July filter	2,238	18.1	18.0
Removed by July filter	914	15.2	94.9
All original items	3,152	17.3	40.3

Table 5: Per-item accuracies use the same original predictions, without rerunning either model. The headline scores, 17.0/42.0, average six operation-count subtasks. “Reset” and “Carry” denote combined recipes, not a one-factor ablation.

moved group; accuracy on retained items is almost unchanged.

All 914 removed items have “nothing” as the correct answer. The carry model selects it on 867 of them (94.9%). One item describes seven boxes, including “Box 6 contains nothing,” then asks “Box 6 contains . . .” Its answers are “nothing,” “the suit,” “the bread,” “the medicine,” and “the egg.” A rule that chooses “nothing” whenever offered succeeds on the entire removed group without tracking any contents. The increase is therefore compatible with a preference for that answer form. The audit identifies this confound; it does not establish why the training recipe changed the preference or whether answer length caused it.

Fresh evaluation on the revised full set of 6,780 items gives 17.8 for reset and 18.8 for carry. SH-10M scores 18.8 and GPT-2 scores 19.1. None exceeds the nominal five-choice chance level of 20.0. Revised fast scores are 18.4 for both recipes. These are new evaluations with subtask averaging, distinct from the retained-item partition in Table 5.

The revised local evaluation includes an index-alignment fix to the convenience scorer: it must

filter gold items in the same way as the dataloader filters predictions. The patch and upstream revision accompany the reproduction code. The issue was reported to the organizers; leaderboard scoring uses a server-side path.

The evidence removes the apparent tracking benefit of the combined recipe. It does not show that resets are intrinsically better or that state carry is useless. We chose per-window resets for the submission because the extra machinery had no demonstrated benefit in these evaluations. The comparison remains single-seed, and small revised differences should not be interpreted as a reliable ranking.

6 Discussion

StateHead complements BabyLM work on larger recurrent models by showing that a small vector state can support many BLiMP contrasts without positional embeddings. BLaLM’s similar qualitative separation between BLiMP and EWoK/COMPS suggests that weak semantic results are not unique to this cell, while differences in data and model size prevent a stronger cross-paper conclusion.

The experiments also delimit what can be attributed to memory. On revised entity tracking, the transformer fails alongside StateHead. These data cannot establish that diagonal recurrence causes the failure. Explicit key–value structure remains a plausible way to improve binding, consistent with research on associative recall and copying (Schlag et al., 2021; Arora et al., 2024; Jelassi et al., 2024). Testing that explanation requires a task and training regime in which a suitable control succeeds.

Sequence length also needs task-specific interpretation. Salhan et al. show that longer training contexts do not uniformly improve reading prediction. Whether longer windows would improve StateHead remains an open question requiring a direct ablation. Lee et al.’s hybrid results suggest benefits from explicit retrieval at larger scales, but do not predict how much benefit would survive the Strict-Small budget. These comparisons motivate controlled follow-up experiments rather than supply explanations established by our data.

7 Conclusion

StateHead shows that a 9.97M-parameter recurrent language model can exceed the official 98.4M GPT-2 checkpoint on BLiMP after approximately 1.1

hours of laptop training. Its recurrent state contains only 2,160 values, and its elementwise update contributes a small fraction of arithmetic operations. The capability profile is uneven: reading-time prediction is poor, and several semantic and tracking evaluations remain near chance. The recipe comparison further shows why a large score difference must be checked at the item level. The resulting submission is a reproducible point of comparison for studying which language capabilities fit within a very small persistent memory.

Limitations

All runs use one seed; the 3.0-point BLiMP margin therefore has no estimate of across-seed variation. Development used fast benchmark subsets, so model and recipe selection were influenced by them. The baseline was evaluated as released rather than retrained with the same tokenizer and optimizer. The two main StateHead configurations differ in more than parameter count, and the earlier reset/carry recipes also change initialization and packing. None of these comparisons isolates an architectural cause.

Historical entity trajectories use the original benchmark and cannot be compared directly with revised full-set results. The item audit identifies an answer-form confound rather than the mechanism by which models acquired their preferences. The older SH-98M GLUE score of 60.0 used a pre-fix fine-tuning pipeline and is omitted, as is its older AoA evaluation. Submission GLUE results use our MLX implementation; every task has not been independently replicated with the official PyTorch trainer.

Head-wise Muon, dropout, training objective, and classification pooling were not separately ablated. Near-zero reading-time performance remains unexplained. Arithmetic and memory figures are analytic estimates with explicit storage conventions. Wall-clock time is measured only for StateHead and does not establish a speedup over GPT-2. Experiments cover one English corpus and one recurrent family, and do not establish long-context accuracy or scaling behavior.

Acknowledgements

The author thanks the BabyLM organizers and reviewers. OpenAI Codex assisted with literature search, text revision, figures, and verification scripts. The author retains full responsibility.

References

- Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. 2024. [Zooology: Measuring and improving recall in efficient language models](#). In *International Conference on Learning Representations*.
- BabyLM Organizing Committee. 2026. [BabyLM 2026 evaluation pipeline](#). Software repository. Revision 7d0a6c8, 8 July 2026.
- Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. 2024. [xLSTM: Extended long short-term memory](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 107547–107603. Curran Associates, Inc.
- Guy E. Blelloch. 1990. [Prefix sums and their applications](#). Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University.
- James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2017. [Quasi-recurrent neural networks](#). In *International Conference on Learning Representations*.
- Tyler A. Chang, Catherine Arnett, Abdelrahman Sadallah, et al. 2025. [Global PIQA: Evaluating common-sense reasoning across 100+ languages and cultures](#). *arXiv preprint arXiv:2510.24081*.
- Tyler A. Chang and Benjamin K. Bergen. 2022. [Word acquisition in neural language models](#). *Transactions of the Association for Computational Linguistics*, 10:1–16.
- Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candace Ross, Raj Sanjay Shah, Alex Warstadt, Ethan Gotlieb Wilcox, and Adina Williams. 2025. [Findings of the third BabyLM challenge: Accelerating language modeling research with cognitively plausible data](#). In *Proceedings of the First BabyLM Workshop*, pages 399–420. Association for Computational Linguistics.
- Lucas Georges Gabriel Charpentier and David Samuel. 2024. [GPT or BERT: Why not both?](#) In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 262–283. Association for Computational Linguistics.
- Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Jaap Jumelet, Tal Linzen, Aaron Mueller, Suchir Salhan, Raj Sanjay Shah, Alex Warstadt, and Ethan Gotlieb Wilcox. 2026. [BabyLM turns 4 and goes multilingual: Call for papers for the 2026 BabyLM workshop](#). *arXiv preprint arXiv:2602.20092*.
- Morten H. Christiansen and Nick Chater. 2016. [The now-or-never bottleneck: A fundamental constraint on language](#). *Behavioral and Brain Sciences*, 39:e62.
- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. [Transformer-XL: Attentive language models beyond a fixed-length context](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988. Association for Computational Linguistics.
- Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. 2024. [Griffin: Mixing gated linear recurrences with local attention for efficient language models](#). *arXiv preprint arXiv:2402.19427*.
- Andrea Gregor de Varda, Marco Marelli, and Simona Amenta. 2024. [Cloze probability, predictability ratings, and computational estimates for 205 English sentences, aligned with existing EEG and reading time data](#). *Behavior Research Methods*, 56(5):5190–5213.
- Jeffrey L. Elman. 1990. [Finding structure in time](#). *Cognitive Science*, 14(2):179–211.
- Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. [Colorless green recurrent networks dream hierarchically](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1195–1205. Association for Computational Linguistics.
- Patrick Haller, Jonas Golde, and Alan Akbik. 2025. [Sample-efficient language modeling with linear attention and lightweight enhancements](#). In *Proceedings of the First BabyLM Workshop*, pages 175–191. Association for Computational Linguistics.
- Awni Hannun, Jagrit Digani, Angelos Katharopoulos, and Ronan Collobert. 2023. [MLX: Efficient and flexible machine learning on Apple silicon](#). Software.
- Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. [Findings of the second BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora](#). In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 1–21. Association for Computational Linguistics.
- Anna A. Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi U. Kumar, Setayesh Radkani, Thomas H. Clark, Carina Kauf, Jennifer Hu, R. T. Pramod, Gabriel Grand, Vivian C. Paulun, Maria Ryskina, Ekin Akyürek, Ethan G. Wilcox, Nafisa Rashid, Leshem Choshen, Roger Levy, Evelina Fedorenko, Joshua Tenenbaum, and Jacob Andreas. 2025. [Elements of world knowledge \(EWoK\): A cognition-inspired](#)

- framework for evaluating basic world knowledge in language models. *Transactions of the Association for Computational Linguistics*, 13:1245–1270.
- Pavel Izmailov, Dmitrii Podoprikin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. [Averaging weights leads to wider optima and better generalization](#). In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence*. AUAI Press.
- Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. 2024. [Repeat after me: Transformers are better than state space models at copying](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 21502–21521. PMLR.
- Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. 2024. [Muon: An optimizer for hidden layers in neural networks](#). Online technical note.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. [Scaling laws for neural language models](#). *arXiv preprint arXiv:2001.08361*.
- Najoung Kim and Sebastian Schuster. 2023. [Entity tracking in language models](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3835–3855. Association for Computational Linguistics.
- Hyunji Lee, Wenhao Yu, Hongming Zhang, Kaixin Ma, Jiyeon Kim, Dong Yu, and Minjoon Seo. 2025. [Understanding and enhancing Mamba-Transformer hybrids for memory recall and language modeling](#). In *Proceedings of the First BabyLM Workshop*, pages 380–398. Association for Computational Linguistics.
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). In *International Conference on Learning Representations*.
- Eric Martin and Chris Cundy. 2018. [Parallelizing linear recurrent neural nets over sequence length](#). In *International Conference on Learning Representations*.
- Kanishka Misra, Julia Rayz, and Allyson Ettinger. 2023. [COMPS: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models](#). In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2928–2949. Association for Computational Linguistics.
- Ofir Press and Lior Wolf. 2017. [Using the output embedding to improve language models](#). In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 157–163. Association for Computational Linguistics.
- Suchir Salhan, Richard Diehl Martinez, Zébulon Goriely, and Paula Buttery. 2025. [What is the best sequence length for BabyLM?](#) In *Proceedings of the First BabyLM Workshop*, pages 130–146. Association for Computational Linguistics.
- Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. 2021. [Linear transformers are secretly fast weight programmers](#). In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9355–9366. PMLR.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. [Neural machine translation of rare words with subword units](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725. Association for Computational Linguistics.
- Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. [SuperGLUE: A stickier benchmark for general-purpose language understanding systems](#). In *Advances in Neural Information Processing Systems*, volume 32, pages 3266–3280.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. [GLUE: A multi-task benchmark and analysis platform for natural language understanding](#). In *International Conference on Learning Representations*.
- Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjape, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. [Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora](#). In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34. Association for Computational Linguistics.
- Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. [BLiMP: The benchmark of linguistic minimal pairs for English](#). *Transactions of the Association for Computational Linguistics*, 8:377–392.
- Ronald J. Williams and Jing Peng. 1990. [An efficient gradient-based algorithm for on-line training of recurrent network trajectories](#). *Neural Computation*, 2(4):490–501.

A Reproducibility

The submitted model, tokenizer, and milestone revisions are available on [Hugging Face \(haybales/statehead-10m\)](#). The [code repository](#) contains training and evaluation implementations, configurations, tokenizers, and evaluator patches. The [original review snapshot](#) is also available.

The PyTorch export supports the Hugging Face causal-language-model interface with remote model code enabled. Reproduction requires the model-specific configuration and tokenizer rather than current repository defaults. The camera-ready source includes scripts and input hashes for the item audit and FLOP cross-check. Preserve the original training recipes and evaluator revision when reproducing the submitted results.

B FLOPs and Memory Accounting

For L layers of width d , vocabulary size V , and MLP expansion r , the dominant per-token forward cost is

$$F_{\text{SH}} \simeq L[(10 + 4r)d^2 + (4\alpha + 4)d] + 2dV. \quad (3)$$

Per layer, the four gate projections cost $8d^2$, the output projection $2d^2$, and the MLP $4rd^2$. The vocabulary projection incurs computation even with tied embeddings. We approximate each nonlinear activation by $\alpha = 4$ operations. The counting tool additionally includes GELU and normalization costs: $L(\alpha rd + 8d) + 4d$. Bias additions and nonlinear-kernel details are not exact hardware operation counts. Vocabulary-output softmax and training loss are outside the forward-logit comparison.

Analytic totals are 19,974,240 and 197,478,432 FLOPs/token for SH-10M and SH-98M. Gate activations and recurrence contribute 43,200 and 178,560, respectively. Full attention adds about $4Td$ matrix-multiplication FLOPs/token/layer, plus a smaller softmax term. The GPT-2 estimate uses 12 layers, width 768, vocabulary 16,384, and 1,024 learned positions.

Profiler cross-check. We instantiated the preserved PyTorch exports with random weights and profiled forward logits on CPU in fp32, using 32 tokens for SH-10M and eight for SH-98M. PyTorch 2.12.1 with `with_flops=True` reports 19,921,680 and 197,191,248 supported FLOPs/token. These differ from the analytic totals by 0.27% and 0.15%,

respectively. The matrix-product counts agree exactly. The profiler does not count every element-wise operator, and this check does not measure the repeated work or runtime of the MLX scan kernel.

Training is estimated as three times forward cost per token, giving $3FD$ for D tokens. The runs process exactly 161,198,080 and 163,196,928 tokens. This approximates forward and backward arithmetic, excluding optimizer operations, memory traffic, and system energy.

One sequence requires Ld recurrent values. BF16 storage occupies $2Ld$ bytes: 4,320 for SH-10M and 17,856 for SH-98M. FP32 storage doubles these amounts. The full-attention cache has $2LTd$ values. These counts exclude parameters, training activations, and temporary workspace.

C Context-Length Development Runs

Development runs used windows of 512, 1,024, and 2,048 tokens with the same 98,658,120-parameter configuration and 16k tokenizer. Accumulation factors were 64, 32, and 16, preserving 32,768 tokens/update. The runs used cosine decay, a 4,650-update horizon, and approximately 152.2M token exposures each. All used flat row packing, no separators, and per-window resets, unlike the main document-carry SH-98M probe.

Original fast evaluations included the subsequently removed entity items. Applying the July filter retrospectively to the saved final predictions gives retained-item accuracies of 17.1, 15.9, and 16.5 for the three lengths, respectively. These are item averages from old predictions, not fresh evaluations under the revised pipeline. They provide no evidence that increasing training length to 2,048 rescued entity tracking in those runs. They also do not establish an optimal length for the submitted 10M model; that requires a controlled evaluation on its own recipe and replication across seeds.

D Full Milestone Trajectories

Milestone	SH-10M				SH-98M			
	BLiMP	Sup.	EWoK	Ent.	BLiMP	Sup.	EWoK	Ent.
1M	51.3	50.0	49.5	44.1	54.1	46.4	48.4	43.1
2M	54.4	54.0	47.5	43.1	54.3	55.6	49.1	43.4
3M	53.6	55.6	47.8	43.3	57.7	55.6	50.1	40.5
4M	55.0	56.8	51.5	42.9	58.6	55.2	50.4	41.4
5M	56.6	57.6	50.2	40.1	60.9	58.8	50.1	29.3
6M	57.9	57.2	49.1	32.7	61.5	58.4	50.1	21.6
7M	59.0	55.6	51.4	40.9	63.4	56.4	49.9	17.2
8M	59.5	55.6	49.3	28.1	63.6	58.0	48.1	41.4
9M	59.7	57.2	50.6	31.9	64.2	54.0	49.9	37.5
10M	60.3	56.8	48.9	40.3	65.1	56.4	50.4	41.0
20M	65.2	53.6	48.1	35.2	70.5	54.0	49.7	38.0
30M	67.4	55.6	50.1	35.4	72.1	60.4	50.2	38.6
40M	68.2	54.8	49.6	38.6	72.6	58.4	49.5	35.7
50M	69.0	51.6	50.0	37.8	72.3	59.2	49.7	34.3
60M	68.8	54.4	51.2	35.6	71.5	56.8	49.1	36.2
70M	68.6	56.4	51.8	37.4	70.1	59.6	51.3	31.2
80M	68.7	57.6	49.5	38.0	68.9	58.4	50.1	27.5
90M	68.0	56.8	49.5	38.0	67.5	54.8	51.5	25.0
100M	67.8	58.8	51.1	36.0	67.0	54.0	50.7	24.7

Table 6: All 19 milestones on the original fast subsets. Labels denote nominal cumulative word exposures scheduled by optimizer updates. Entity scores predate the July filter and are historical measurements, not revised-benchmark tracking results. The 98M columns refer to individual checkpoints.