

Hard-Wiring the Induction Head: The Surprising Effectiveness of Copy Mechanisms in Data-Efficient Language Modelling

Josh Taylor

British Standards Institution

<https://www.linkedin.com/in/josh-taylor-24806975/>

Abstract

Large language models need several orders of magnitude more text than a child hears to reach a comparable level of linguistic skill. One reason is that a particularly useful capability, the *induction head* (a circuit that completes a pattern of the form $[A][B] \dots [A]$ by copying $[B]$), is not built in but must emerge during training, and does so only after the model has read a great deal of text. We ask whether supplying this capability directly, as a fixed part of the architecture, improves learning when data is scarce. We present a three-layer, attention-free language model of about 12M parameters in which all long-range mixing is performed by a non-learned, sub-quadratic induction operator (its cost grows more slowly than the square of the sequence length): for each position it finds recent earlier occurrences of the same token and copies the continuation of the best match. Trained on the 10M-word BabyLM Strict-Small corpus, the model learns markedly faster than a parameter-matched self-attention baseline and stays competitive at convergence; on the public BabyLM evaluation it is competitive with the challenge’s GPT-2 baselines on grammatical minimal pairs with a tenth of the parameters. Targeted interventions show the operator is used as intended: predictions travel along its links, the links are necessary, and stacking layers composes them into two-step retrieval. Because the operator forms no attention matrix, it is also cheaper at length: on a single GPU it overtakes causal Flash attention at about 1,000 tokens of context and is $3\times$ faster and $9\times$ smaller in memory at 8,192, where attention approaches the memory ceiling. These results motivate hybrid designs that interleave exact look-back with attention.

1 Introduction

Children learn to understand and produce language after hearing on the order of tens of millions of words. Modern language models reach comparable

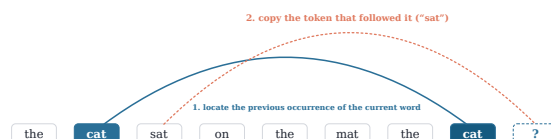


Figure 1: The fixed look-back-and-copy rule. To predict the token after the second “cat,” the operator finds the previous “cat” and copies the token that followed it.

fluency only after training on three to four orders of magnitude more text (Warstadt et al., 2023). The BabyLM Challenge (Warstadt et al., 2023; Charpentier et al., 2025) turns this contrast into a benchmark by fixing a child-sized training budget; we work in its STRICT-SMALL setting, roughly 10 million words.

Part of the gap comes from models having to rediscover useful machinery for themselves. A clear example is the *induction head*, a circuit first identified by Elhage et al. (2021) and analysed by Olsson et al. (2022). Given a repeated pattern $[A][B] \dots [A]$, an induction head predicts $[B]$ by finding the earlier occurrence of $[A]$ and copying what followed it. In a standard transformer (Vaswani et al., 2017) this circuit is absent at the start; it appears abruptly partway through training, only once the model has seen substantial data (Olsson et al., 2022). Because the same circuit underpins much of a model’s ability to reuse information from earlier in its input, its late arrival is costly when data is scarce.

If a mechanism is this useful and this well understood, there is no obvious reason to make every model learn it from scratch. We supply it directly: we replace the learned long-range mechanism with a fixed operator that performs the look-up and copy explicitly (Figure 1), and ask whether this helps at BabyLM scale.

All code, training scripts and evaluation har-

nesses are publicly available.¹

Contributions. (1) An attention-free, sub-quadratic architecture combining a fixed induction operator for long-range copying with a dynamic convolution for local mixing. (2) Evidence that, on a 10M-word budget, this model is substantially more data-efficient than a parameter-matched self-attention model, remains competitive at convergence, and holds its own against the challenge’s GPT-2 baselines on the public benchmark at a tenth of the size. (3) Targeted interventions showing the operator is genuinely used and composes across layers into two-step retrieval. (4) A measured efficiency comparison against causal Flash attention across context lengths, locating the crossover at about 1,000 tokens. (5) An account of the alternative designs that failed on the way. (6) A direction for hybrids that interleave exact look-back with attention.

2 Background

Attention and its cost. Self-attention (Vaswani et al., 2017) computes, for every position, a weighted average over all other positions. Because every position is compared with every other, computation and memory grow with the square of the sequence length.

Induction heads. Induction heads (Elhage et al., 2021; Olsson et al., 2022) are responsible for much of a transformer’s in-context copying. A key property is that the circuit needs at least two layers: one to attach to each token a record of what preceded it, and a second to use that record to copy forward (Olsson et al., 2022).

Efficient sequence mixing. Dynamic convolutions (Wu et al., 2019) predict a position-specific filter from the current token and apply it to a local window, scaling linearly with length. Long implicit convolutions extend the filter across the sequence (Poli et al., 2023). Linear attention (Katharopoulos et al., 2020) and state-space models such as Mamba (Gu and Dao, 2023) summarise the past in a compact running state. A consistent recent finding is that this family struggles precisely on *recall*: fixed-size-state architectures have difficulty retrieving specific earlier tokens, on synthetic tests and real language (Arora et al., 2024) and on explicit copying (Jelassi et al., 2024). Our design takes that seriously: the local mixer follows the dynamic-

convolution approach, while the long-range mixer stores no compressed state at all and instead looks up exact token matches.

For the remaining components we use standard choices: a gated feed-forward block of the SwiGLU family (Shazeer, 2020) and rotary position embeddings (Su et al., 2021).

3 Architecture

The model has three identical layers, each applying in order: a dynamic depthwise convolution for local mixing; the induction operator for exact long-range copying; and a gated feed-forward block. It has about 12M parameters, width 384, and a vocabulary of 8,000 subword units learned from the training text. Figure 2 shows the design.

The induction operator. For each position, the operator finds the most recent earlier positions holding the same token (exact match on token identity; the five most recent matches are kept as candidates). A small learned scoring function compares the context around the current position with the context around each candidate and ranks them; the token that followed the best-ranked candidate is copied forward, without averaging over candidates. Two properties matter. The operator is *not learned*: the look-up is computed directly from the token sequence, so it works from the first training step; only the ranking is trained. And it is *cheap*: matches are found by sorting token identities, so cost grows as the sequence length times its logarithm rather than its square, and no attention matrix is formed.

The division of labour is deliberate: the convolution handles immediate neighbours, the operator handles exact reuse at any distance, and the feed-forward block transforms the result. As Section 5.3 shows, local context written by the convolution is also carried along the induction links, so the two mixers cooperate rather than duplicate each other.

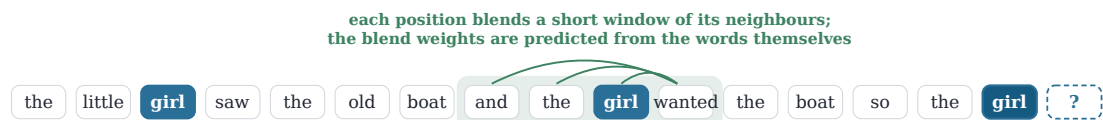
4 Experimental setup

Data. We train on the BabyLM STRICT-SMALL corpus (Warstadt et al., 2023), about 10M words of child-directed and child-accessible English, tokenised into sequences of 256 tokens. Total training exposure is about 98 million tokens, roughly six passes over the corpus, within the challenge’s ten-pass limit.

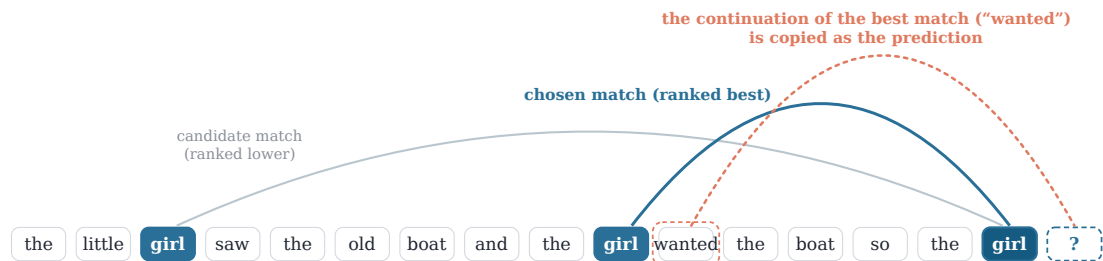
Models. We compare against a baseline identical in every respect except that the induction operator is replaced by multi-head self-attention (Vaswani

¹<https://github.com/joshua-taylor/conv-induction-babyLM>

A Local mixing (dynamic convolution)



B Exact look-back (induction operator)



C One layer in detail (the model stacks three)

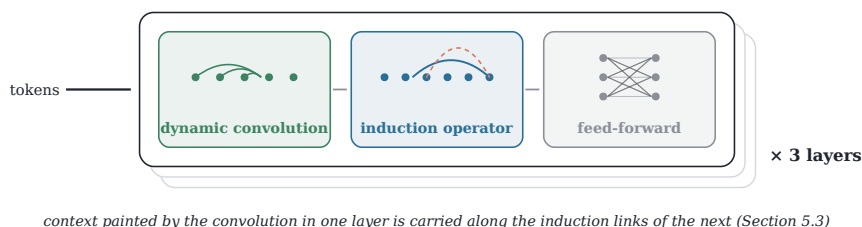


Figure 2: The full model. (A) A dynamic convolution blends each position with a short window of neighbours, with blend weights predicted from the words themselves. (B) The induction operator gathers recent earlier occurrences of the current token as candidates, ranks them, and copies the continuation of the best match; here the third “girl” looks back to two earlier “girl” positions and copies “wanted.” (C) One layer in detail; the model stacks three. Context written by the convolution in one layer is carried along the induction links of the next.

et al., 2017). The two models are parameter-matched (12.3M learnable weights each), so differences reflect the mechanism, not capacity. Each is trained with three random seeds.

Reporting. Our main measure is validation perplexity (how surprised the model is by held-out text; lower is better). To measure copying directly we build sequences whose second half repeats their first half and score how often the model predicts the repeated continuation. Benchmark scores in Section 5.5 come from the official BabyLM evaluation pipeline.

5 Results

5.1 Data efficiency and competitiveness

Figure 3 (left) tracks validation perplexity as each model sees more text. Early in training our model is far ahead: after about 10M words its perplexity is 57, against 78 for the attention baseline, which closes the gap only after roughly ten times as much text. The right panel explains why. On the copying probe our model succeeds from the start (accuracy

around 0.5), whereas the attention model barely learns to copy, rising from almost zero to about 0.05 even after 100M words. The fixed operator hands the model a working copy mechanism immediately; the attention model has to grow one and, at this scale, largely fails to.

At convergence the attention model reaches slightly lower perplexity (31.9 vs. 34.3, three seeds, spread about 0.5; Table 1). This is the expected price of a fixed mechanism: given enough data, a fully learned component is more flexible. The point we draw is that a non-learned, sub-quadratic long-range operator keeps the model within about seven percent of a strong attention baseline while making it considerably faster to learn from scarce data.

5.2 What each component contributes

We removed the components one at a time, re-trained, and measured the rise in perplexity (Figure 4). This is a deletion test rather than a parameter-matched comparison, but the pattern is

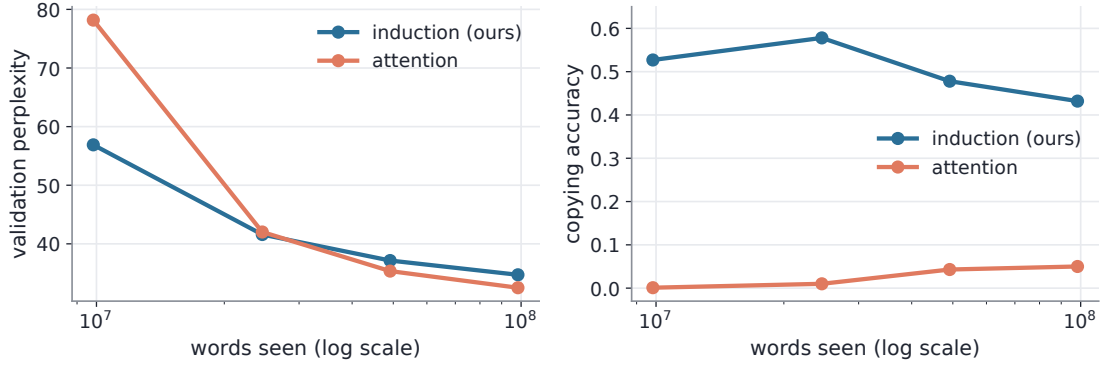


Figure 3: Left: our model is far more data-efficient early and is overtaken only after about ten times more text. Right: our model copies from the outset; the attention model barely learns to within the budget.

Model	Params	Perplexity ↓
Induction operator (ours)	12.3M	34.3
Self-attention baseline	12.3M	31.9

Table 1: Parameter-matched comparison at the full budget, averaged over three seeds (spread ~ 0.5).

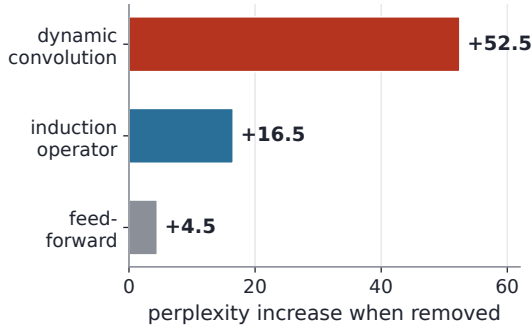


Figure 4: Removing the convolution is most damaging; removing the induction operator clearly hurts; removing the feed-forward block hurts least, despite holding the most parameters.

informative. Removing the convolution is most damaging (34 to 87); removing the induction operator clearly hurts (to 51); removing the feed-forward block costs least (to 39) despite it holding the largest share of parameters. The purpose-built mixers, not raw capacity, contribute most to model performance.

5.3 Is the operator used as intended?

A built-in tool is only worthwhile if the model relies on it. We ran four interventions. Throughout, a *link* means one use of the operator: the connection from a position to the earlier occurrence it reads from.

Predictions travel along the links. For each prediction we identified the exact earlier position

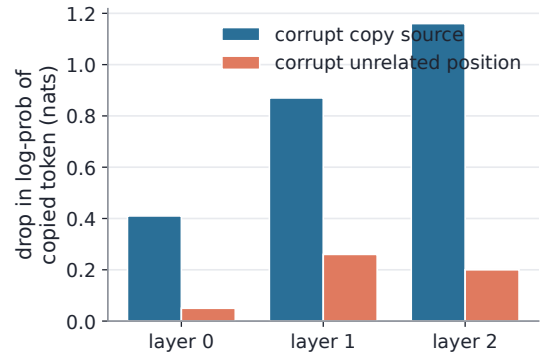


Figure 5: Corrupting the position a prediction copies from is far more damaging than corrupting an unrelated position the same distance back, and the effect grows with depth.

it was copying from and corrupted the model’s internal representation there, replacing it with an uninformative average. The model’s confidence in the copied token fell sharply; corrupting an unrelated position the same distance back changed almost nothing (Figure 5). The effect grows from early to late layers, indicating the copy is assembled towards the end of the network. This held across more than two thousand cases.

The links are necessary. The first intervention corrupted the source; the second removes the link itself. For each prediction we identified the candidate the operator relied on most, then deleted it from the operator’s candidate list, so the model could no longer read from that position through the operator while everything else was left intact. Confidence in the correct token fell by about three nats across two hundred cases (a nat is the natural unit of log-probability; three nats means the assigned probability falls by a factor of about twenty). As a control, on the predictions where one existed we

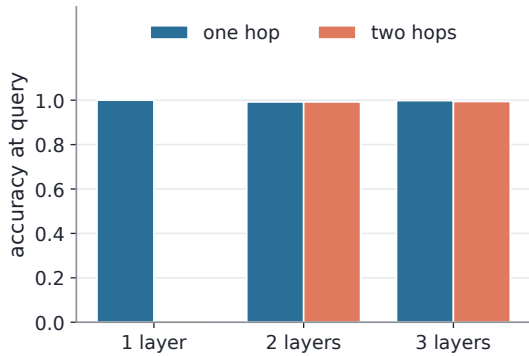


Figure 6: One layer solves a single look-up but not a two-step chain; two layers solve both.

instead deleted a rarely-used candidate whose following token differed from the answer: this cost essentially nothing, while deleting the real link on the same predictions cost about four nats. Removing the link that matters collapses the prediction; removing another changes nothing. Most strong long-range links sat on genuinely repeated spans of text, indicating the operator responds to real repetition rather than coincidence.

Stacking layers enables two-step retrieval. We built a synthetic test in which the answer can only be reached in two steps: the query pattern points to an intermediate token, and the answer is the continuation of that intermediate token’s own earlier occurrence. A one-layer model handles a single look-up perfectly but fails the chain, returning the intermediate token instead; a two-layer model solves both (Figure 6), matching the known two-layer requirement for induction circuits (Olsson et al., 2022). In practice, chained look-ups are a primitive form of combining separated information. A reader does this when a passage says the captain owns the boat and, later, that the boat is called the Maria: connecting “captain” to “Maria” requires passing through “boat.” A single look-up can only re-surface what sat next to a cue; chaining lets facts stated in different places be combined, and our fixed operator gains this ability exactly when layers are stacked.

What the links carry. The copied representation identifies the source token almost perfectly (about 0.98, by a linear read-out). It also carries the source’s immediate left-hand context, but only when the convolution is active: the word to the left of the source is recoverable 76% of the time with the convolution on, against 16% with it off. The convolution writes local context onto a token; the

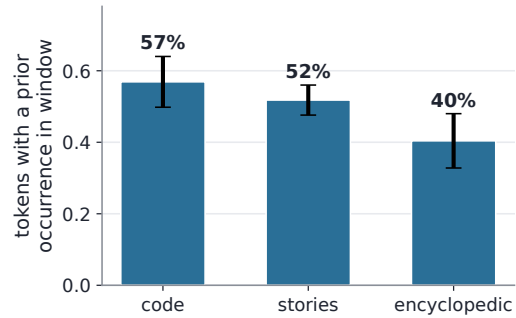


Figure 7: Token repetition is common in every genre examined; this is the structure the operator exploits.

operator transports the package across the gap.

How much is free. A rule with no learning, copying the continuation of the most recent match, already gets 23% of these predictions right; the trained model reaches 31%. Most of the copying ability is supplied free by the fixed operator, with learning adding a useful but secondary refinement in the ranking.

5.4 The pattern in natural text

The approach pays off only if real text repeats itself. It does: across three kinds of writing, 40–57% of tokens have already appeared earlier in the same 256-token window (Figure 7). Code repeats most (57%), children’s stories next (52%), encyclopedic prose least (40%), though its repeats reach back furthest. Appendix A visualises the operator’s strongest links on a sample sentence: they land on repeated content words, tying each occurrence to its earlier mention.

5.5 Benchmark evaluation

We evaluated with the official BabyLM pipeline and submitted to the public leaderboard. Table 2 compares against the challenge’s official baselines: two GPT-2 entries, standard 124M-parameter transformers (Radford et al., 2019) roughly ten times our size, and the GPT-BERT baselines, which combine masked and causal training and were the strongest systems of the previous challenge. The metrics: BLiMP and its supplement test grammar by asking whether the model prefers a grammatical sentence over a minimally different ungrammatical one (Warstadt et al., 2020); EWok tests basic world knowledge the same way; entity tracking tests whether the model follows the state of objects through a passage; COMPS tests knowledge of concept properties; (Super)GLUE is a suite of

	BLiMP	BLiMP Sup.	EWoK	Entity Tr.	COMPS	GLUE
Ours (attention-free, 12M)	65.7	55.5	51.6	27.6	50.7	61.8
GPT-2 (strict-small)	63.1	57.3	—	21.1	51.8	63.8
GPT-2 (10M baseline)	66.4	57.1	49.9	13.9	—	55.9
GPT-BERT (causal)	71.7	63.2	49.5	34.6	—	65.1
GPT-BERT (mntp)	71.3	63.3	49.9	40.0	—	65.7

Table 2: Public BabyLM leaderboard scores against the official baselines (higher is better; bold marks the best score in each column; a dash means that evaluation is not reported on the leaderboard for that entry). The GPT-2 entries are standard 124M-parameter transformers (Radford et al., 2019); the GPT-BERT baselines combine masked and causal training, and we show the strongest of their six configurations under each scoring mode (mntp = masked next-token prediction scoring). On the minimal-pair tasks chance is 50, and all models at this scale sit near it on the knowledge-heavy tasks (EWoK, COMPS). We omit the leaderboard aggregate and age-of-acquisition metric, which depend on a fuller checkpoint series than we uploaded; the columns shown are computed identically for all models.

language-understanding tasks on which the model is fine-tuned.

Three things stand out. First, an attention-free model whose only long-range mechanism is fixed is competitive with the GPT-2 baselines on the core grammar test: our BLiMP score of 65.7 sits between the two GPT-2 entries (63.1 and 66.4) at a tenth of their parameters, and we are well ahead of both on entity tracking (27.6 vs. 21.1 and 13.9), the task closest in spirit to what the operator provides: following a specific thing through a passage. Second, the GPT-BERT baselines remain clearly stronger on grammar and entity tracking; they combine masked and causal training, and interleaving our operator into such models (Section 8) is a natural way to test whether the two advantages add. Third, on the knowledge-heavy tasks every model sits near chance; a 10M-word diet contains little encyclopedic knowledge, whatever the architecture.

6 Efficiency

The long-range operator forms no attention matrix: matches are found by sorting token identities, so computation and memory grow as the sequence length times its logarithm rather than as its square. To measure what this is worth we timed two three-layer stacks that are identical except for the sequence mixer, one using the induction operator and one using causal self-attention through PyTorch’s fused (Flash) kernel, on a single NVIDIA T4 at batch size 4 in half precision, building the index once per forward pass as the trained model does.

Figure 8 shows the result. At the 256-token training length attention is about $1.8\times$ faster: the quadratic term is still small and the fused kernel is well optimised, while our operator pays a fixed cost for sorting and gathering. The two cross at roughly 1,000 tokens, after which the gap widens,

reaching $1.3\times$ at 2,048 tokens, $1.8\times$ at 4,096 and $3.1\times$ at 8,192. Memory is the starker and more practical difference: peak usage grows close to linearly for the operator, from 0.10 GB at 256 tokens to 1.14 GB at 8,192, against 10.43 GB for attention at that length, a $9\times$ gap. At 16,384 tokens attention exhausts the GPU while the induction stack completes the forward pass in 2.20 GB, so on this hardware the model can process contexts the parameter-matched attention model cannot run at all. These are forward-pass measurements at one model width, and the advantage belongs to long context: it is invisible at the 256-token sequences the BabyLM budget calls for.

7 What did not work

The final design is simple, but the route to it was not; we arrived at the exact-match operator largely by elimination. Throughout, we followed a fixed discipline: change one variable at a time, write down the predicted outcome and an abandonment criterion before running, and stop when the criterion was met.

Compressed-state mixers could not do exact recall. Our first long-range mixers summarised the past in a fixed-size state, in the spirit of linear attention. They trained stably but plateaued well above the attention baseline, failing exactly where retrieval had to be precise, mirroring the recall gap documented for this family (Arora et al., 2024; Jelassi et al., 2024). Squeezing history into a fixed-size summary blurs precisely the information an induction step needs.

A learned associative memory collapsed into linear attention. A memory storing key–value pairs in superposition in a single vector was abandoned under the criterion we had written down in advance: its behaviour became indistinguishable

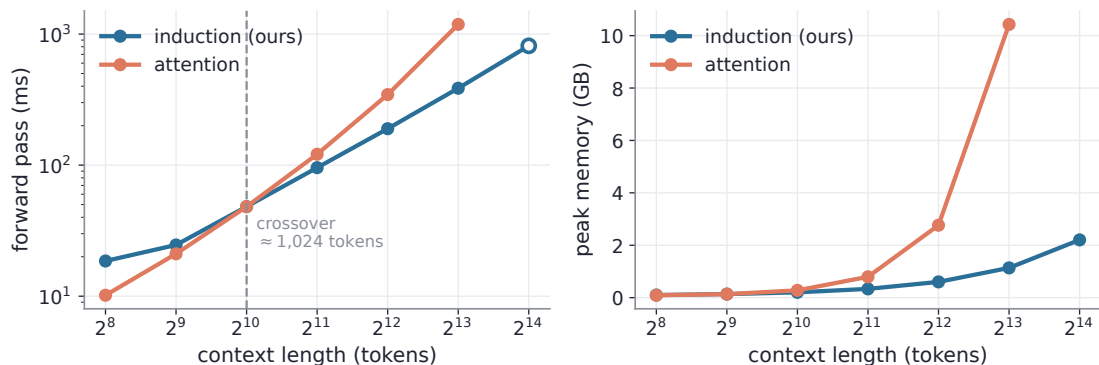


Figure 8: Measured cost against context length for two three-layer stacks differing only in the sequence mixer (NVIDIA T4, batch 4, half precision). Left: forward-pass latency, log-log; the crossover is at about 1,000 tokens, and the open marker at 16,384 is the induction stack completing a length at which attention runs out of memory. Right: peak memory, the more practical limit.

from the linear attention it was meant to improve on. New mathematics over a compressed state does not change the state’s fundamental limit.

Counting statistics could not replace the learned ranking. Variants choosing among candidates by frequency or recency alone handled raw copying but fell to near chance on context-dependent grammar such as subject–verb agreement. Choosing which occurrence to trust requires comparing contexts, and that comparison must be learned; the final design keeps learning exactly there and nowhere else in the long-range path.

Fixed convolution filters underperformed content-dependent ones. Convolutions with filters fixed after training, in the spirit of long implicit-filter designs (Poli et al., 2023), were consistently worse than the dynamic form (Wu et al., 2019). At this scale, local mixing must adapt to content.

The search was broad before it was narrow. Beyond the failures above, we implemented and abandoned reservoir-style mixers with large fixed random recurrent states; purely count-based predictors; attention over compressed keys and values; temporal convolutional and trellis-network backbones; and Fourier-transform-based mixing. Each received the same treatment and fell to one of two failure modes: it compressed away the exact retrieval that language rewards, or it added flexibility a 10M-word budget could not train. The breadth of the search is part of the evidence: at this scale, one precise, almost-free mechanism beat every cleverer alternative we tried.

8 Future directions

Interleaving with attention. Exact look-back and attention are complementary. Attention spreads a normalised weight distribution across all positions, so the weight on any one distant token is diluted over long contexts: flexible but smoothed routing. The operator provides a precise pointer to one exact position. Interleaving induction and attention layers would save parameters and give attention an absolute look-back it does not naturally have, a second, sharper route for information. Hybrids that add a precise retrieval path to an efficient backbone recover most of the recall gap to attention (Arora et al., 2024), supporting the direction.

Longer context. Section 6 shows the operator is already cheaper than fused attention beyond about a thousand tokens, and can run contexts that exhaust attention on the same hardware. Its natural home is therefore long-context modelling. Training at those lengths, rather than only measuring the forward pass, is the obvious next step; the BabyLM budget gave us no reason to train beyond 256 tokens.

9 Limitations

At the full budget the attention baseline reaches slightly lower perplexity, so the contribution is data efficiency rather than an outright win. In the component study (Section 5.2), removing a component also removes its parameters, so those comparisons measure the value of a component, not of a mechanism at fixed size. In Section 5.3, the comparison between cutting the most-used link and cutting an unused one was only possible on the small number of predictions that had a suitable unused link, so those particular numbers carry more uncertainty

than the headline result. The two-step result uses a synthetic task: it shows the mechanism can chain, not that natural text requires chaining. Benchmark differences on the knowledge-heavy tasks sit near chance for all models at this scale and should not be over-read. Finally, the efficiency measurements in Section 6 are forward-pass timings of one model width on one GPU, and the advantage only appears beyond about a thousand tokens: at the 256-token sequences this model is trained on, attention is the faster of the two. We have not trained at long context, so the efficiency result speaks to the cost of the mechanism, not to the quality of a long-context model built from it.

10 Conclusion

Building the induction head in, rather than waiting for it to emerge, makes a small language model markedly more data-efficient while keeping it competitive with a parameter-matched attention model, and it holds its own against transformer baselines ten times its size on the public BabyLM grammar tests. The operator is genuinely used, necessary, and composes across layers into two-step retrieval; it rests on repetition, a pattern that pervades real text; and being attention-free it is also cheaper than fused attention beyond about a thousand tokens of context, running lengths that exhaust attention on the same GPU. Some capabilities are worth supplying directly when data is scarce, and an exact look-back may complement attention rather than merely replace it.

References

- Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. 2024. Zoology: Measuring and improving recall in efficient language models. In *International Conference on Learning Representations*. ArXiv:2312.04927.
- Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Y. Hu, Jing Liu, Jaap Jumelet, Tal Linzen, Aaron Mueller, Candance 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*, Suzhou, China. Association for Computational Linguistics.
- Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, et al. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*.
- Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.
- Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. 2024. Repeat after me: Transformers are better than state space models at copying. In *International Conference on Machine Learning*. ArXiv:2402.01032.
- Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*.
- Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, et al. 2022. In-context learning and induction heads. *Transformer Circuits Thread*. ArXiv:2209.11895.
- Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. 2023. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*. ArXiv:2302.10646.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8).
- Noam Shazeer. 2020. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*.
- Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2021. RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*.
- Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, 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*, Singapore. 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.

Felix Wu, Angela Fan, Alexei Baevski, Yann N. Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *International Conference on Learning Representations*. ArXiv:1901.10430.

A Qualitative look-back links

Figure 9 shows the operator’s strongest links on a sample sentence, one row per layer, with darker and thicker arcs marking links the model relies on more. The strong links land on repeated content words such as “old,” “boat,” “girl” and “river.”

B Model and training details

Each of the three identical layers applies, in order, with pre-norm residual connections: (1) a gated dynamic depthwise convolution, in which a small network predicts, from each token, the weights of three short depthwise filters (kernel size 3, dilations 1, 2 and 4, 16 weight-sharing groups) whose outputs are summed and gated; (2) the induction operator, which indexes, for every position, the five most recent earlier positions holding the same token (computed exactly, without learning, by sorting token identities), scores each candidate with a learned multi-head comparison between the query context and the candidate context, and copies forward the representation at the position immediately after the best-scoring candidate, with a learned sink allowing the operator to abstain; and (3) a SwiGLU feed-forward block. Token order is encoded with learned position embeddings up to 512 positions. Table 3 lists the configuration.

Hyperparameter	Value
Layers	3
Model width	384
Attention-equivalent heads	4
Feed-forward width	1,536
Vocabulary (byte-level BPE)	8,000
Sequence length	256
Max position embeddings	512
Conv kernel / dilations / groups	3 / 1,2,4 / 16
Induction candidates per position	5
Parameters	12.3M
Batch size	32
Training steps	12,000 (~6 epochs)
Optimiser: 2D matrices	Muon, lr 0.03
momentum / NS steps	0.95 / 5
Optimiser: other tensors	AdamW, lr 6×10^{-4}
Weight decay / grad clip	0.1 / 1.0
Schedule	300-step warmup, cosine to 5%
Checkpoint selection	BLiMP proxy

Table 3: Architecture and training configuration.

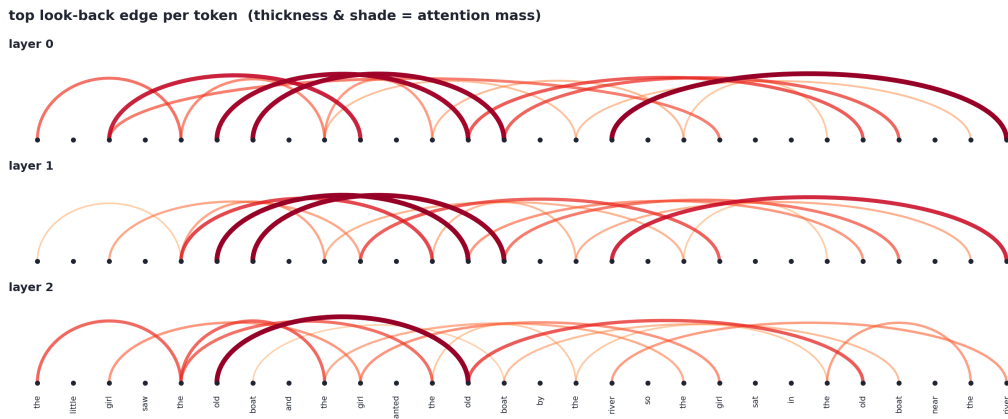


Figure 9: The operator’s strongest links on a sample sentence (one row per layer); thicker, darker arcs mark links the model relies on more.