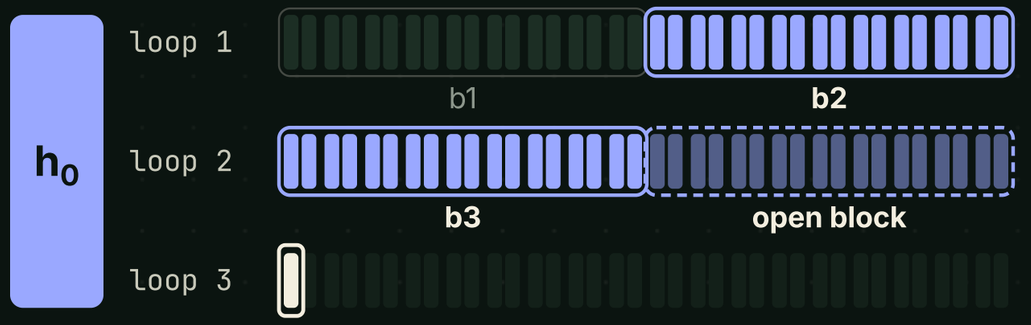
CCAI AGENTS SEMINAR, 14:00

TUESDAY, 17 NOVEMBER 2026

ARLoop

attention over depth *for looped transformers*



ARLoop-0.6B drawn from its configuration: 3 loops of the same 18 layers. In blue, what the first attention sublayer of loop 3 reads.

A review of the paper, its released checkpoints re-measured, and its idea tested where every answer is checked by a proof checker.

Bartosz Naskręcki

Faculty of Mathematics and Computer Science, Adam Mickiewicz University in Poznań

Paper: Yuekun Yao, Yupei Du, Dawei Zhu, Huan Sun, *ARLoop: Fully-Open Pre-trained Looped Transformers with Attention Residuals*, arXiv:2611.05234, submitted 2026-11-03.

slides, notes, report
bnaskrecki.faculty.wmi.amu.edu.pl/loop

I



PART I

The idea: *attention over depth*

Does a looped model use its extra loops? ARLoop says the residual design decides it, and changes that design.

I THE IDEA
II THE PAPER'S EVIDENCE
III RELEASED CHECKPOINTS · E1
IV CHECKED PROOFS · E2, E3, DEMO
V TRAINING MORE · WHAT WE FOUND

Does an extra loop get used? *In our own runs, rarely.*

In our project `latent`, language models with recurrent depth write proofs that a checker accepts. What their extra loops did:

96,127

decisions of a learned stopping rule: **every one** chose a single loop

129 · 141 · 124

theorems proved, of 192, with 1, 2 and 4 forced loops: no steady gain

128 vs 131

proved, of 192: a learned gate that adds a loop per theorem, against always one loop

12 → 10

checked answers, of 70, of a tiny model with 1 loop, then with 2 or more

ARLOOP'S CLAIM

The residual design decides how much the later loops are used.

→

ITS TEST

A released matched pair: ARLoop-0.6B and PreNormLoop-0.6B, the same 180B tokens; only the residual path differs.

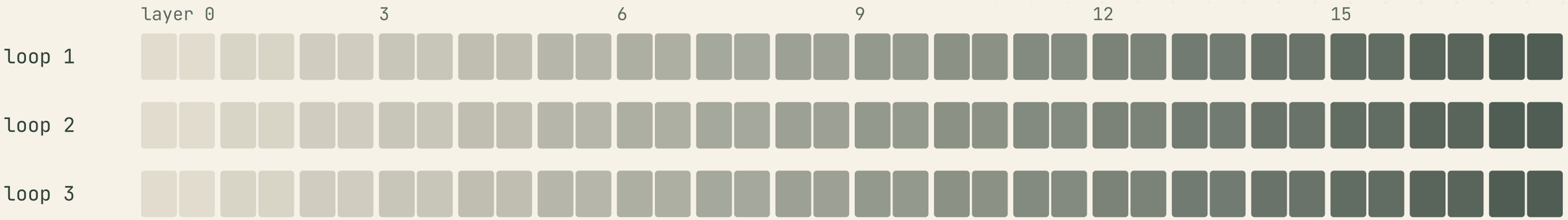
→

OUR ANSWER, IN SHORT

After long training, more loops fix single proof steps; whole proofs, not yet (Part V).

From latent's research map: campaign 7 (the stopping rule and the forced loops), a separate gate run, and the canary's run 284986. Context: an arXiv scan for looped models, 2026-08-07 to 2026-11-15, gave 412 hits.

A looped transformer runs *the same layers again*



18	×	3	=	54
shared layers (the columns)		loops (the rows)		effective layers: more depth, no new layer weights

WHAT MAKES THE LOOP TRAINABLE: ALWAYS THE PATH FROM ONE LOOP TO THE NEXT

Huginn	**Ouro**	**Parcae**	**DeepLoop**	**ARLoop**
Geiping et al. 2025	Zhu et al. 2025	Prairie et al. 2026	Li et al. 2026	Yao et al. 2026
input injection, sandwich norm	a learned exit	a decaying transition	residual scaling	attention over depth

BOARD · PANELS 1 AND 2: THE SUM, THEN THE SOFTMAX

ARLoop replaces PreNorm's fixed sum by *a softmax over depth*

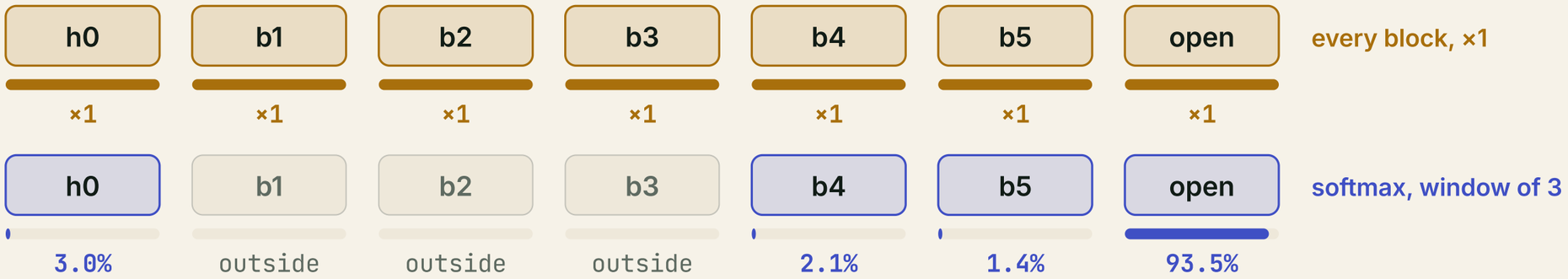
$$h_i = \sum_{j=0}^{i} \alpha_{ij}\, z_j, \qquad \alpha_{i\cdot} = \mathrm{softmax}_j\left(q_i^\top \mathrm{Norm}(z_j)\right)$$

z_j output of sublayer j; z_0 the embedding

q_i a learned routing query of sublayer i

α_{ij} weights, positive, summing to 1

PreNorm: every α_{ij} fixed at 1, not learned

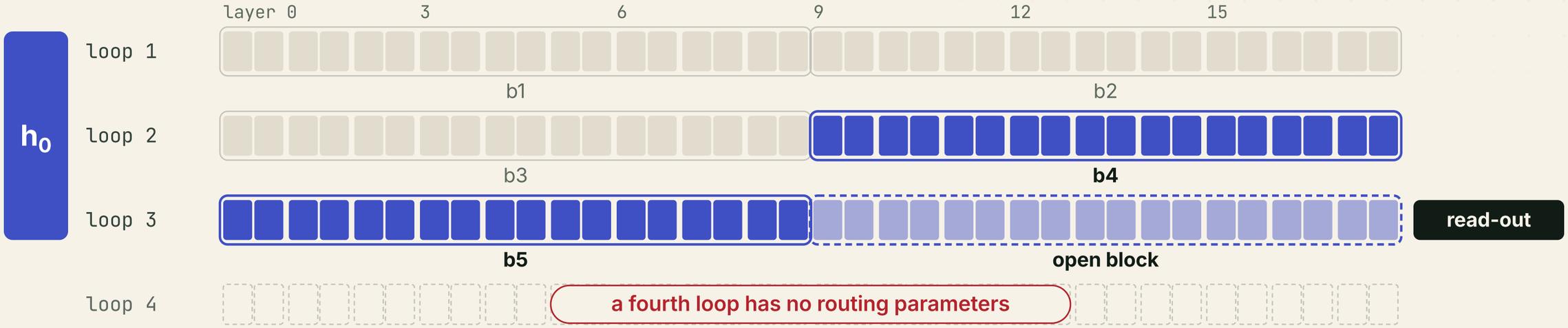


THE IDEA

In PreNorm every earlier update stays in the sum with weight 1; only later updates can counteract it. ARLoop learns the weights: its read-out puts **93.5%** on the most recent block.

Read-out weights (the last routing call) of the released 0.6B models, measured by our routing probe (E1). The idea comes from Kimi's Attention Residuals (arXiv:2603.15031).

Routing reads *a window of blocks*, with its own parameters in every loop



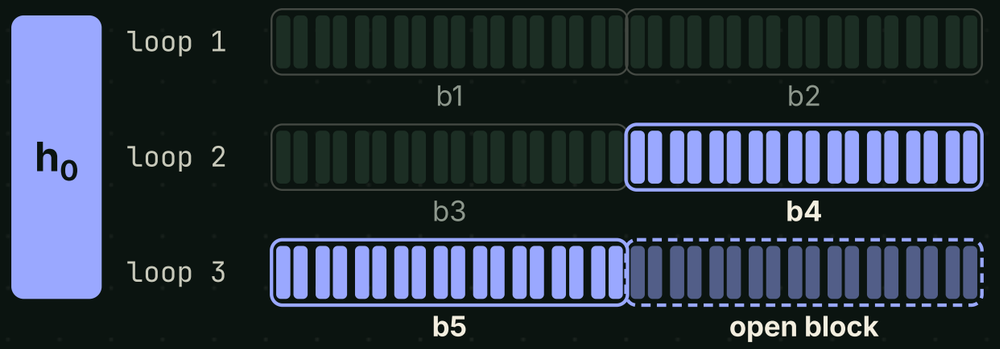
334,848

routing parameters, a query and a norm per sublayer: exactly ARLoop-0.6B minus PreNormLoop-0.6B

F1 The layers are shared across loops; the routing is not. There are no routing parameters for a fourth loop, so running ARLoop with more loops than trained needs a rule for them. PreNorm needs none.

A block is 9 layers; the window is the embedding and the 3 most recent blocks (outlined). Parameters: 583,138,944 – 582,804,096 (configuration, code, Hugging Face totals).

II



PART II

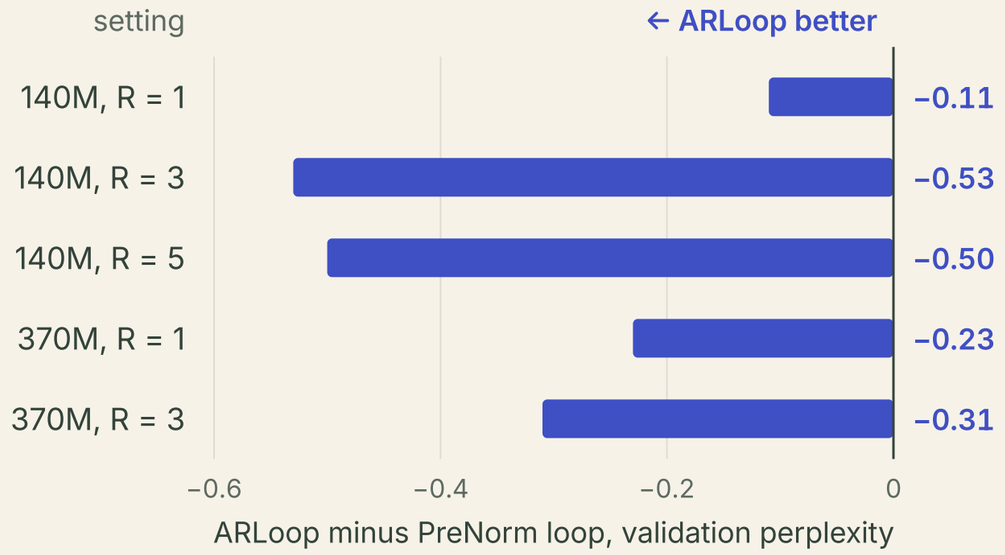
The paper's evidence

A controlled study of six residual designs, a matched pair of released models, and what the evidence does not cover.

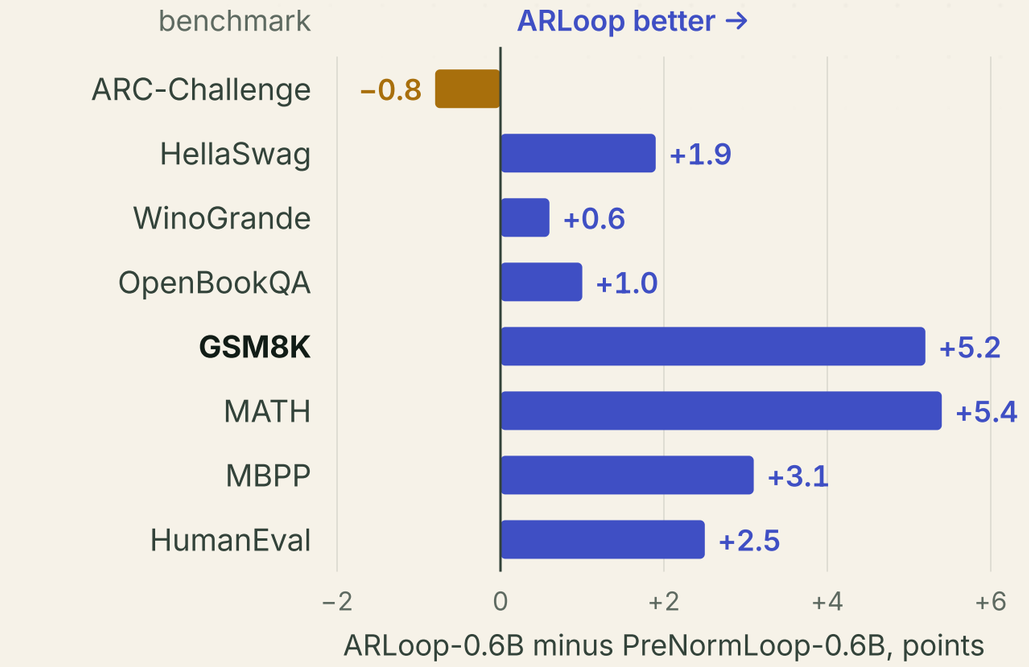
I THE IDEA
II THE PAPER'S EVIDENCE
III RELEASED CHECKPOINTS · E1
IV CHECKED PROOFS · E2, E3, DEMO
V TRAINING MORE · WHAT WE FOUND

In the paper, ARLoop wins *every perplexity setting and 7 of 8 tasks*

CONTROLLED STUDY: SIX DESIGNS, SAME DATA (TAB. 1)



RELEASED 0.6B PAIR: SAME 180B TOKENS (TAB. 3)



CAVEATS F7 One run per cell: no seeds, no intervals; the gain is not monotone in R. F2 Both 0.6B models saw GSM8K-style data late in training: fair within the pair, not against other models.

Tab. 1: validation perplexity, 10M FineWeb-Edu tokens; ARLoop is the lowest of all six designs, by 0.11 to 0.23 over the next best. Tab. 3: single checkpoints.

What the evidence *does not cover* PAPER

WINDOW NEEDS VALUE RESIDUALS F11

18.20 vs PreNorm's **18.19**

perplexity of the window without value residuals (140M, R = 3): no better than PreNorm. The main text calls the window enough.

NO TEST BEYOND THE TRAINED LOOPS F4

$R \leq 4$

gains stop at the trained number of loops, shown only for small full-history models, never for the released ones. That is our E1.

FIXED BEATS RANDOM DEPTH

16.07 vs **16.69**

perplexity at R = 4 trained with a fixed against a random number of loops (12.01 vs 12.33 at 370M). `latent` trained with random depth.

COST

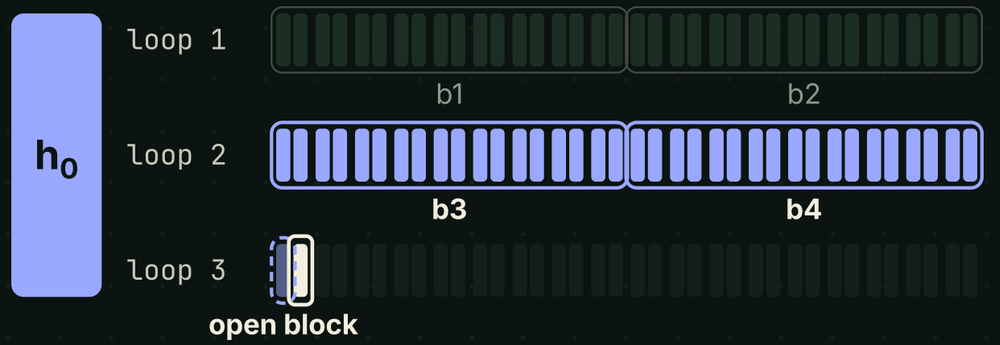
−10%

training throughput with the window at R = 4 on one H100 (28.9k → 25.9k tokens/s); the same memory as PreNorm.

SO WE TEST **The released models were never run beyond their trained depth.** That is our first test, E1, together with the headline gap on GSM8K.

Paper: Tab. 2, App. A.1, A.3 (window); App. D.2 (extrapolation); App. D.1 (random depth: Poisson, clamped to 1–4 loops, mean 3.24); Tab. 4 (cost).

III



PART III · E1

The released checkpoints, *re-measured*

Does the headline gap hold up? Do the released models gain from more loops than they were trained with?

I THE IDEA
II THE PAPER'S EVIDENCE
III RELEASED CHECKPOINTS · E1
IV CHECKED PROOFS · E2, E3, DEMO
V TRAINING MORE · WHAT WE FOUND

The GSM8K gap reproduces; *without the BOS token it is inconclusive*

RELEASED MODELS
ARLoop-0.6B, PreNormLoop-0.6B, ARLoop-1.2B, pinned revisions

→

GSM8K, AS THE AUTHORS RUN IT
all 1,319 items, 5-shot chain of thought, greedy, BOS token prepended

→

R = 1 … 6 LOOPS
beyond 3, ARLoop reuses loop 3's routing (repeat) or loop 1's (cycle)

→

PAIRED TEST
the same items: bootstrap 95% interval, exact McNemar; rules fixed in advance

+5.16 points

ARLoop-0.6B minus PreNormLoop-0.6B, GSM8K at R = 3, paired over 1,319 items: 95% interval [+2.65, +7.66]; solved by ARLoop only **176**, by PreNorm only **108**, exact McNemar $p = 6.5 \times 10^{-5}$

GSM8K, R = 3, %	PAPER	OURS	OURS - PAPER
ARLoop-0.6B	40.9	39.0	−1.9
PreNormLoop-0.6B	35.7	33.8	−1.9
ARLoop-1.2B	63.1	63.0	−0.1

Registered rule for the gap: SUPPORTS

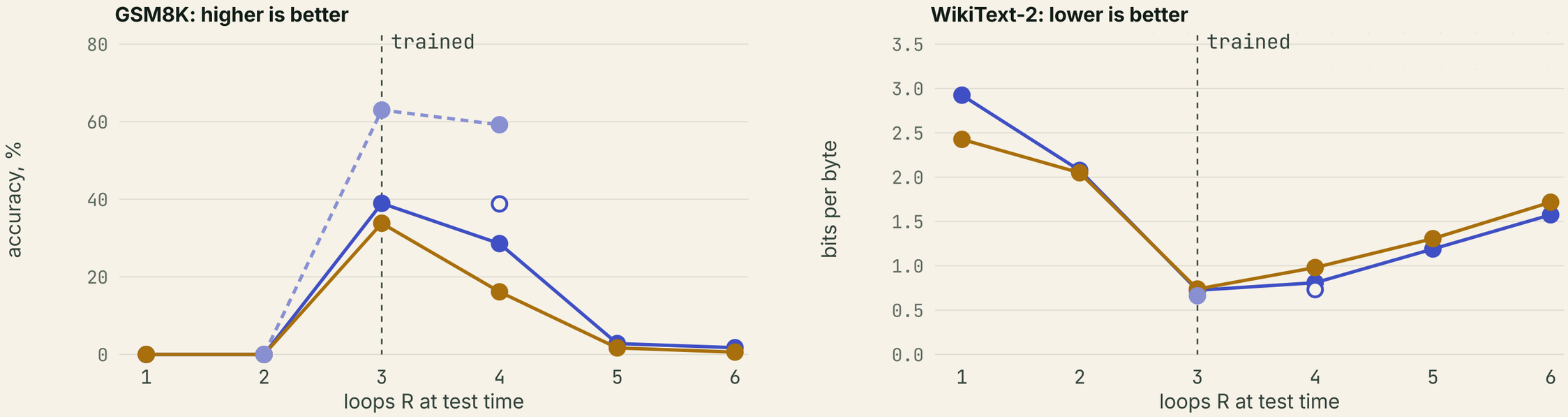
ARLoop-1.2B within ±2 points of 63.1: SUPPORTS

REPRODUCED

The gap reproduces. F9 Without the BOS token: +2.20 [−0.45, +4.85], **inconclusive**. BOS lowers PreNorm by 2.43 points; ARLoop shows no detectable change.

Rules committed before any full run (experiments/e1_release/PROTOCOL.md); our one deviation: the Hugging Face backend, not vLLM. Re-checked from the raw generations.

Below the trained depth, zero accuracy; *above it, no gain detected*

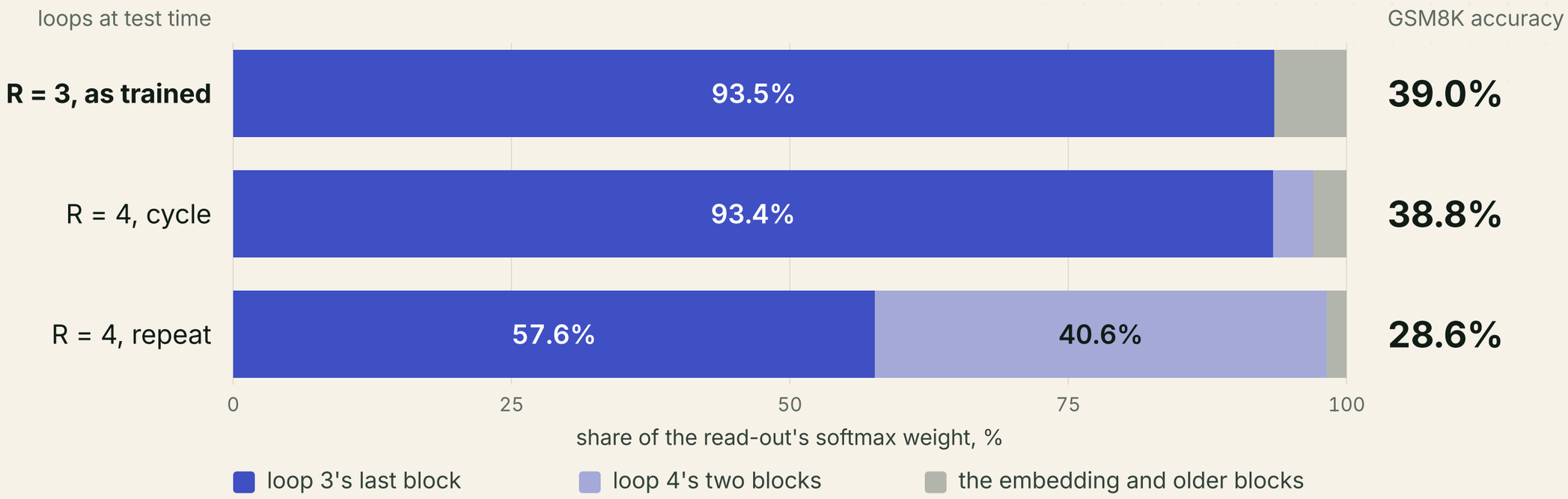


RESULT

Fewer loops than trained: no correct answer. More loops: no gain detected (8 registered tests).
ARLoop's fourth loop: −0.2 [−2.0, +1.7] under cycle, −10.4 points under repeat.

Each R against the same model at R = 3, paired over all GSM8K items; rules for R > 3 "no gain", for R = 2 "R = 3 is better"; re-checked independently.

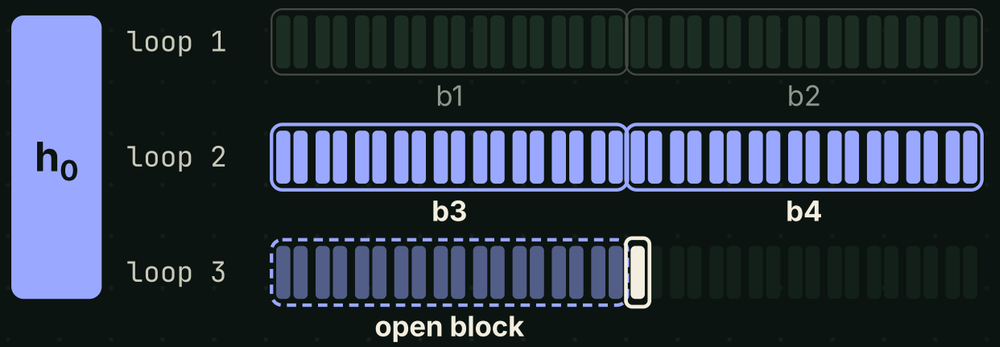
With a fourth loop, *the trained read-out routes around it* EXPLORATORY



EXPLORATORY **Cycle:** the read-out keeps 93.4% on loop 3's last block; accuracy holds. **Repeat:** loop 4 takes 40.6%; accuracy falls 10.4 points (PreNorm, without routing: 17.7 points).

Routing probe on GSM8K prompts, mean over tokens. cycle: loop 4 reuses loop 1's routing parameters; repeat: loop 3's. One probe, not registered, not re-checked.

IV



PART IV · E2, E3

Where every answer is *checked*

Two testbeds from `latent`: a tiny model trained from scratch (E2) and the released pair fine-tuned to write proofs (E3).

I THE IDEA
II THE PAPER'S EVIDENCE
III RELEASED CHECKPOINTS · E1
IV CHECKED PROOFS · E2, E3, DEMO
V TRAINING MORE · WHAT WE FOUND

On the canary, *no new arm beats latent's decoder*

PROMPT
a+b=, 93 training identities

→

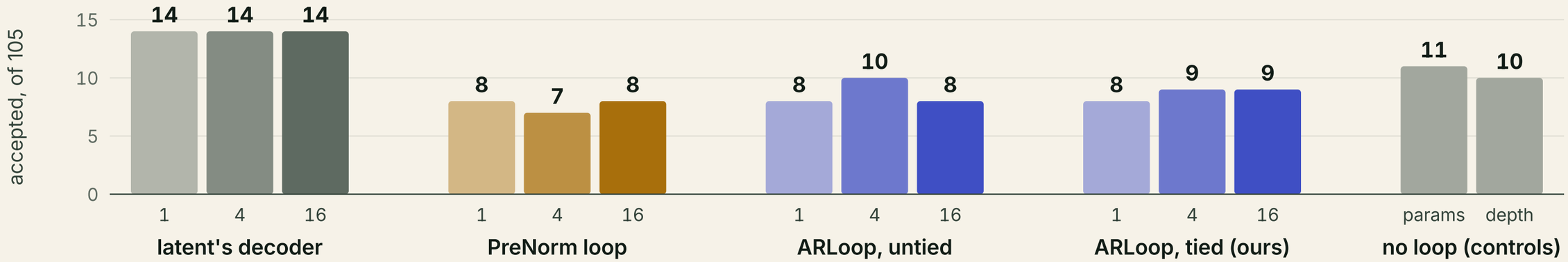
TINY LOOPED MODEL
width 64, 1,000 updates, a random number of loops per batch; seeds 11, 29, 47

→

CERTIFICATE
a compact proof that a+b equals the answer

→

LEAN
accepts or rejects it



arm; under each bar the loops R at test time (in-distribution items, 3 seeds summed)

RESULT

No new arm beats latent's decoder: 6 of 6 primary differences are below 0, none is a candidate effect, and no arm gets a single deeper item right (0 of 192 in every arm and R).

latent's decoder: sandwich norm, input injection; it reproduces latent's run (1,834 of 1,834 records). No arm has value residuals. Seeds 11, 29, 47; recounted independently.

✎ BOARD · PANEL 1, ERASED: THE E3 GRID

After 512 updates, *no proof at any number of loops*

THE PAIR, FROZEN
ARLoop-0.6B and PreNormLoop-0.6B with the same LoRA adapters (rank 32)

→

FINE-TUNE
512 updates on 154 proof sessions, at 3 loops or at a random 1, 2 or 4

→

PROVE
32 goals, one step per call, at R = 1, 2, 3, 4

→

CHECK
a pinned proof checker replays every proof

0 of 32

goals proved in each of the 16 suites: 2 models × 2 trainings × 4 loop counts

32 of 32

gold proofs accepted through the same harness: the checker and the pipeline work

4 of 8 cells

where ARLoop's failed attempts get further than PreNorm's (exploratory, defined after the results)

AT THE FLOOR

No proof at any number of loops. By the rule fixed in advance the comparison decides nothing: the models were trained too little. So we trained longer (Part V).

One seed; at most 96 steps and 600 s per goal; development goals only (the test split stays sealed). Recounted independently from the raw records.

▶ LIVE · GPU NODE python demo.py

Live: one GSM8K question, *both models, R = 1, 2, 3, 4*

ONE QUESTION
a 5-shot prompt, greedy decoding

→

ARLOOP-0.6B
R = 1, 2, 3, 4 (at 4: the repeat rule)

→

PRENORMLOOP-0.6B
R = 1, 2, 3, 4

→

WATCH
the answer, the speed, ARLoop's weight on the embedding

THE DEMO SCRIPT, LINES 49–55

```
for R in [int(x) for x in a.loops.split(",")]:
    set_loops(model, R, "repeat")
    rec = RouteRecorder(model) if is_arloop(model) else None
    torch.cuda.synchronize(); t0 = time.time()
    with torch.inference_mode():
        out = model.generate(ids, max_new_tokens=a.max_new, do_sample=False)
    torch.cuda.synchronize(); dt = time.time() - t0
```

Compare speeds only within one model (ARLoop runs with a routing recorder). "Embedding share" is the softmax weight on the embedding.

OPTIONAL · ONLY IF THE LIVE RUN FAILS

The recorded run: *with fewer loops than trained, no answer*

THE QUESTION (GSM8K, TEST ITEM 0)

Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market? **Correct answer: 18.**

EXTRACTED ANSWER	R = 1	R = 2	R = 3	R = 4
ARLoop-0.6B	no answer	no answer	**9** wrong	**22** wrong
PreNormLoop-0.6B	no answer	no answer	**18** right	**2** wrong

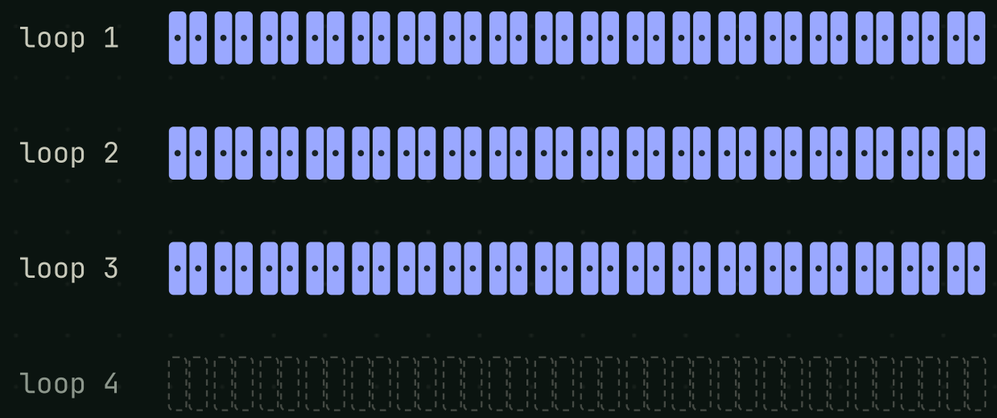
Tokens per second at R = 1, 2, 3, 4: ARLoop-0.6B 36.3, 20.8, 13.6, 10.3; PreNormLoop-0.6B 92.8, 48.0, 32.3, 24.4 (ARLoop runs with a routing recorder: compare speeds only within one model).

ONE QUESTION

One question says nothing about accuracy; the paired test over all items does (E1). It shows the collapse below the trained depth.

The same command, recorded on a GPU node (runs/demo/recorded.txt; the code is experiments/demo/demo.py); the correct answer is GSM8K's own.

V



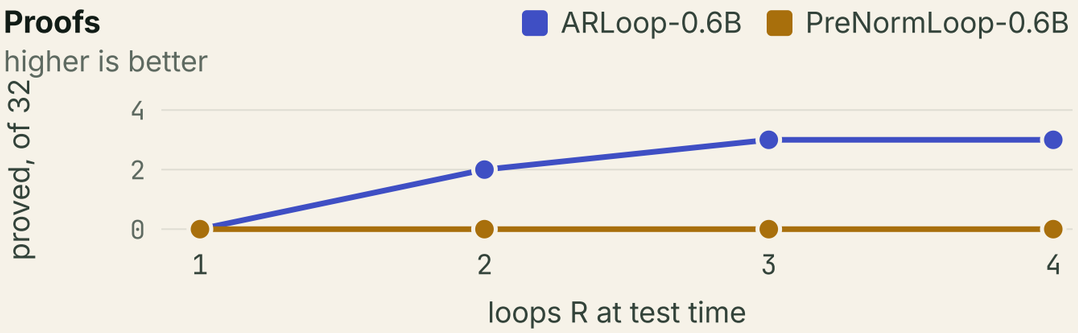
PART V · STAGES 1, 4, 2

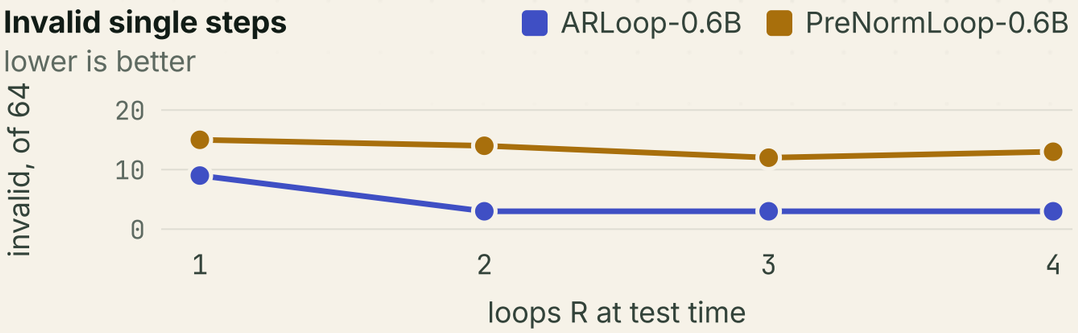
Training more: *does looping start to pay off?*

In E1–E3 the models were trained little, or not at all beyond three loops. So we trained more, with every test fixed in advance.

I THE IDEA
II THE PAPER'S EVIDENCE
III RELEASED CHECKPOINTS · E1
IV CHECKED PROOFS · E2, E3, DEMO
V TRAINING MORE · WHAT WE FOUND

Trained 8× longer: *a one-seed lead for ARLoop*





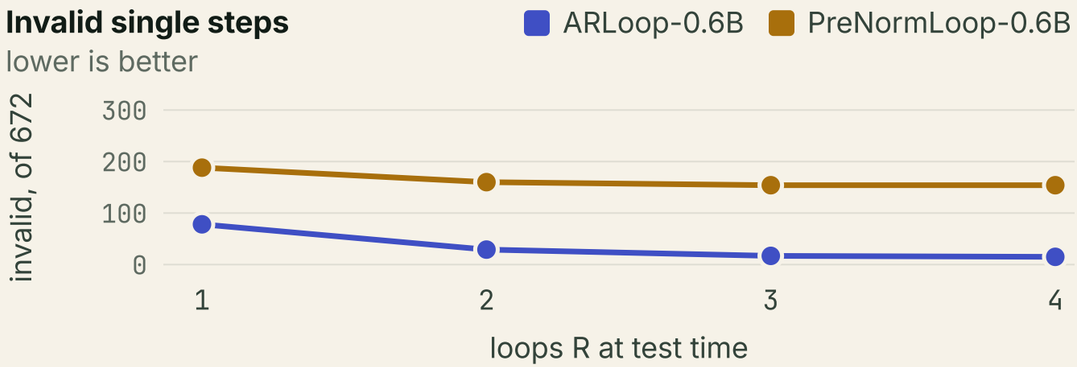
1A · A LEAD	**6 vs 0** steps valid only at 4 loops vs only at 1; Holm p = 0.0625, just above the registered 0.05
1B · NONE DETECTED	the canary trained 10× and 30× longer: **0 of 8** depth tests show an effect
1C · NONE DETECTED	a trained fourth loop of the released ARLoop: **−0.61 points** on GSM8K [−2.20, +1.06]
OFF THE FLOOR	trained and tested at 3 loops: ARLoop proves **4**, PreNorm **3** of 32 goals (after 512 updates: 0 and 0)

NEXT **A one-seed lead is not a result:** Stage 4 re-tests it with three seeds on sealed goals.

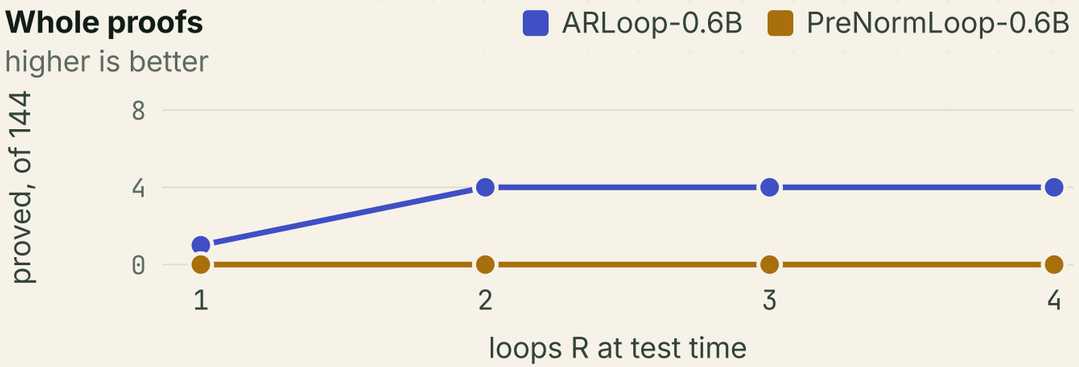
1a: the E3 recipe, 8× the updates (4,096), a random 1, 2 or 4 loops per update, one seed. 1b: 3 seeds. 1c: 1,319 GSM8K items. Each part re-checked; the 30 proofs replayed to QED.

Stage 4: more loops fix *ARLoop's single proof steps*, not yet whole proofs

Invalid single steps
lower is better



Whole proofs
higher is better



Both models trained with a random 1, 2 or 4 loops per update; 3 seeds summed; the test split of latent's proof corpus, opened once.

64 vs 1

Steps (P1): valid only at 4 loops vs only at 1; Holm $p = 7.2 \times 10^{-18}$

4 vs 1

Whole proofs (secondary): proved only at 4 loops vs only at 1; $p = 0.38$

4 vs 5

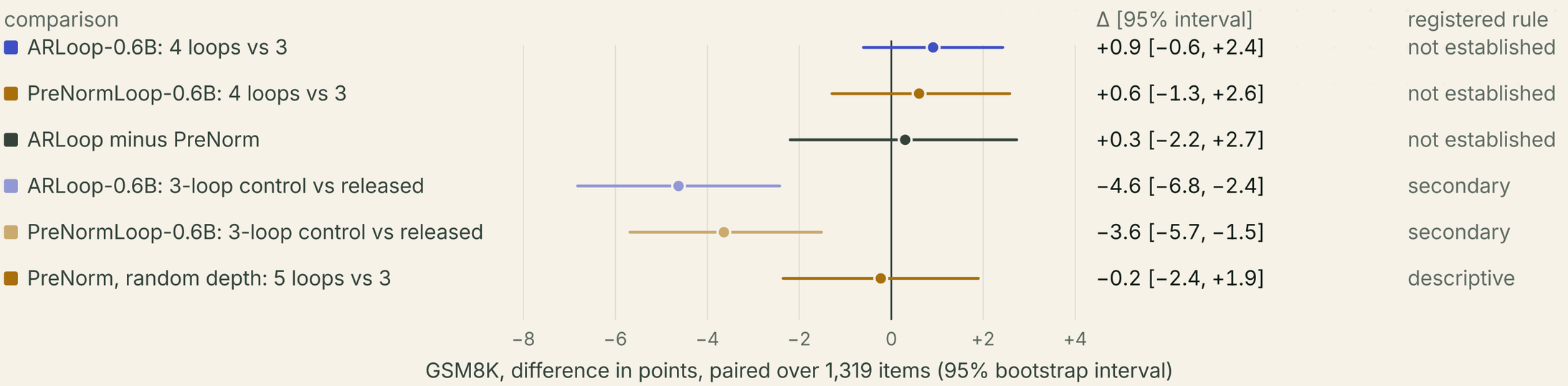
Pair at 3 loops (P3): proved only by ARLoop vs only by PreNorm; none detected

CONFIRMED

Confirmed, 3 of 3 seeds: more loops make ARLoop's single steps valid, more than PreNorm's, which gain too (secondary). **Whole proofs: no detectable change yet.**

Registered before the test split was opened (P1, P3 with Holm; P2 secondary). Recounted independently; the 34 accepted proofs replayed to QED. 48 goals, 224 roster rows.

Stage 2: trained at four loops, *no gain detected*



Every parameter of both released 0.6B models trained further on 0.1B tokens of math text (3,051 updates, one seed): once at 3 loops (the control) and once at 4; PreNorm also with a random number of loops per update, 2 to 5.

NONE DETECTED

No gain detected from training at four loops for either model; the contrast is not established either. The extra training itself cost accuracy at three loops (ARLoop −4.62, PreNorm −3.64 points).

Registered before any run (per model with Holm; the contrast secondary; random depth descriptive). Every item re-scored independently. One seed; each model on its own GPU type.

What we found: *loops can fix single proof steps*, not yet whole proofs

E1	**Does the GSM8K gap hold?**	REPRODUCES	**+5.16 points** [+2.65, +7.66]; without BOS, inconclusive F9
E1	**More loops than trained?**	NONE DETECTED	no gain detected in 8 comparisons; fewer loops: no answer
E2	**Attention over depth, tiny model?**	NONE DETECTED	0 candidate effects in 6 comparisons
E3	**Proofs after a short fine-tune?**	FLOOR	0 of 32 goals in all 16 suites
S1, S4	**Trained 8× longer: valid steps?**	CONFIRMED	ARLoop **64 vs 1** (3 of 3 seeds); PreNorm 51 vs 17
S4	**... and whole proofs?**	NOT YET	ARLoop 4 vs 1 goals, $p = 0.38$
S2	**Whole model trained at 4 loops?**	NONE DETECTED	ARLoop +0.91 [−0.61, +2.43], PreNorm +0.61 [−1.29, +2.58]

NEXT **Does a better step finish a proof?** Continue both models from the same states. Then a task whose loop has a known logical step, and Stage 3: both designs from scratch at 140M.

OPTIONAL · SKIP IF TIME IS SHORT

Reading the paper and its code: *11 findings*

F1 WORDING / IMPLICIT DESIGN Routing has its own query and norm per sublayer: exactly the pair's parameter gap, never counted in the paper. More loops than trained need a rule.

F2 CONTEXT Both 0.6B models saw GSM8K-style data late in training: fair within the pair, not against other models.

F3 WORDING "The only fully-open looped LLM" (Huginn-3.5B) holds only in a narrow sense; Parcae and a recurrent OLMo-2 1B came earlier.

F4 SCOPE The no-extrapolation evidence comes from small full-history models, not from the released ones.

F5 CONFIRMED (WORKS) Two model links redirect to the released names; both work.

F6 WORDING The first block starts from the embedding (code), not from zero (eq. 4).

F7 WORDING "Larger gains at deeper recurrences" (Tab. 1) holds against one loop but is not monotone.

F8 WORDING / IMPLICIT DESIGN The first routing call reads only the embedding: its query and norm are inert.

F9 SCOPE (ROBUSTNESS) The GSM8K gap is clear with BOS (the authors' default), inconclusive without; BOS lowers PreNorm and leaves no detectable change in ARLoop.

F10 IMPLICIT DESIGN Each routing query and its norm act as one product, exactly zero on most coordinates.

F11 SCOPE The main text calls the window enough; the appendix shows it needs value residuals.

Short form; the full text of each finding is in the digest, §10 (in the notes).

References

- Y. Yao, Y. Du, D. Zhu, H. Sun. ARLoop: Fully-Open Pre-trained Looped Transformers with Attention Residuals. arXiv:2611.05234, 2026. Models: osunlp/ARLoop-0.6B, osunlp/PreNormLoop-0.6B, osunlp/ARLoop-1.2B.
- Kimi Team. Attention Residuals. arXiv:2603.15031, 2026.
- J. Geiping et al. Scaling up Test-Time Compute with Latent Reasoning: A Recurrent Depth Approach. arXiv:2502.05171, 2025.
- R.-J. Zhu et al. Scaling Latent Reasoning via Looped Language Models (Ouro). arXiv:2510.25741, 2025.
- H. Prairie, Z. Novack, T. Berg-Kirkpatrick, D. Y. Fu. Parcae: Scaling Laws for Stable Looped Language Models. arXiv:2604.12946, 2026.
- S. Li, Y. Zhang, J. Guo, Q. Gu, M. Wang. DeepLoop: Depth Scaling for Looped Transformers. arXiv:2607.13491, 2026.
- M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, Ł. Kaiser. Universal Transformers. ICLR 2019.
- N. Saunshi, N. Dikkala, Z. Li, S. Kumar, S. J. Reddi. Reasoning with Latent Thoughts: On the Power of Looped Transformers. ICLR 2025.
- R. Csordás, C. D. Manning, C. Potts. Do Language Models Use Their Depth Efficiently? arXiv:2505.13898, 2025.
- S. McLeish et al. Teaching Pretrained Language Models to Think Deeper with Retrofitted Recurrence. arXiv:2511.07384, 2025.
- Z. Zhou, T. Wu, Z. Jiang, F. Obeid, Z. Lan. Value Residual Learning. arXiv:2410.17897, 2025.
- This study: slides, notes and the full report at bnaskrecki.faculty.wmi.amu.edu.pl/loop; the sibling project `latent`.

Thank you.

Slides, the notes with every number computed from the result files, and the full report: **bnaskrecki.faculty.wmi.amu.edu.pl/loop**

`slides` /loop/slides/
`notes` /loop/book/
`report` /loop/report/

slides, notes, report
bnaskrecki.faculty.
wmi.amu.edu.pl/loop

