First Step Toward Reasoning
: State Tracking and Recurrence

추론 능력을 위한 첫 번째 관문: 상태 추적과 회귀

Summer Vacation Toy Project Session · 2026.08.26

YAI 18th NLP Junior Research Team

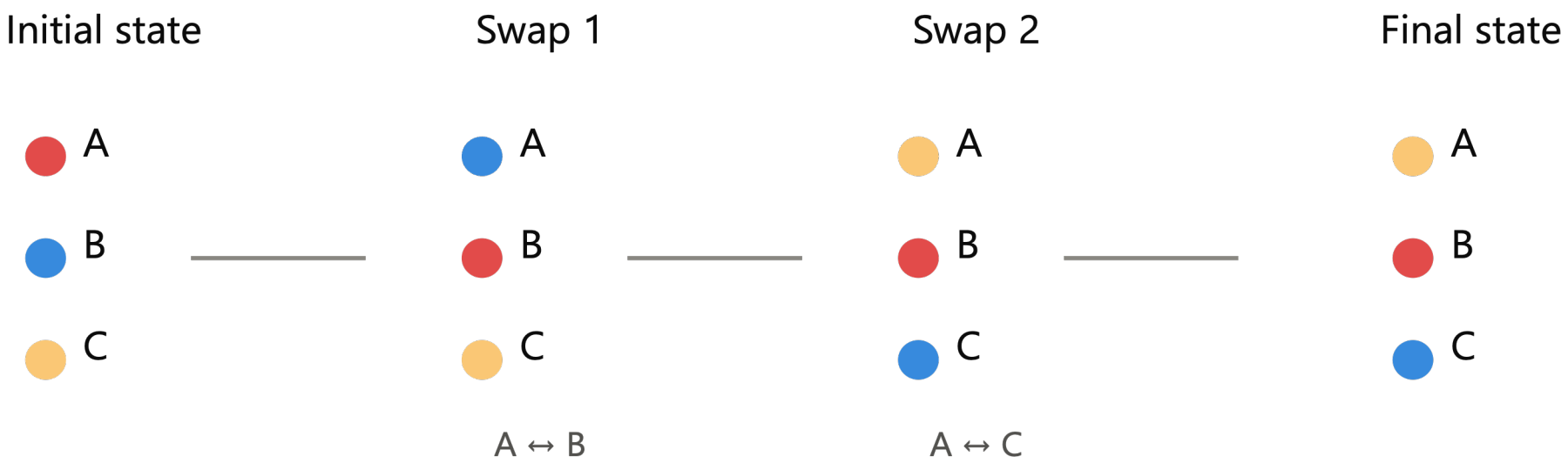
Nayeong Koh, Sunghoon Jung, Yoonsung Jung, Yongjun Choi

Github: https://github.com/avianinfluenza/status-tracking-experiment.git

Contents

Task

- **Ball Swap: predict every person's final color after N ordered swaps**



Task

- **Input = 5 INIT assignments + N ordered SWAP events**
- INIT[i, color] sets each person's starting ball
- SWAP[a, b] exchanges current colors; event order matters
- Five SLOT queries ask for all five final colors
- Primary metric = Exact Match: all five slots must be correct
- **Training = final-state cross-entropy only; no intermediate labels**

```
00:  3 [INIT_0_2] (Yoonseong=yellow)
01:  7 [INIT_1_1] (Seonghun=orange)
02: 11 [INIT_2_0] (Nayoung=red)
03: 20 [INIT_3_4] (Jeongju=blue)
04: 24 [INIT_4_3] (Yongjun=green)
05: 39 [SWAP_2_3] (Nayoung<->Jeongju)
06: 35 [SWAP_1_4] (Seonghun<->Yongjun)
07: 28 [SWAP_0_2] (Yoonseong<->Nayoung)
08: 47 [SWAP_4_1] (Yongjun<->Seonghun)
09: 28 [SWAP_0_2] (Yoonseong<->Nayoung)
10: 28 [SWAP_0_2] (Yoonseong<->Nayoung)
11: 47 [SWAP_4_1] (Yongjun<->Seonghun)
12: 45 [SWAP_3_4] (Jeongju<->Yongjun)
13: 36 [SWAP_2_0] (Nayoung<->Yoonseong)
14: 46 [SWAP_4_0] (Yongjun<->Yoonseong)
15: 51 [SLOT_0] (Yoonseong)
16: 52 [SLOT_1] (Seonghun)
17: 53 [SLOT_2] (Nayoung)
18: 54 [SLOT_3] (Jeongju)
19: 55 [SLOT_4] (Yongjun)
```

Inference?

Inference

60 languages

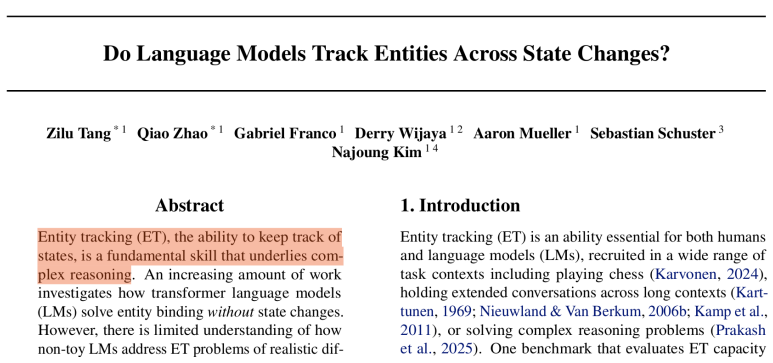
Article Talk Read Edit View history

Wikipedia, the free encyclopedia.

Inference is the act or process of deriving a logical conclusion from known or verified information. Therefore, inference can be described as reaching one judgment based on another.

Why State Tracking?

- **Def. State Tracking**: The ability to maintain prior state and update the current state w/ new information.
- **State Tracking** is fundamental for high-level **reasoning**.
  - *"**Entity tracking** is a fundamental skill that underlies **complex reasoning**."* [1]
  - *"**Entity tracking** is essential for **complex reasoning**."* [2]
  - *'Robust **state tracking** is needed for **complex reasoning**.'* [3]



Do Language Models Track Entities Across State Changes?

Zilu Tang [*1] Qiao Zhao [*1] Gabriel Franco [1] Derry Wijaya [1 2] Aaron Mueller [1] Sebastian Schuster [3] Najoung Kim [1 4]

Abstract

Entity tracking (ET), the ability to keep track of states, is a fundamental skill that underlies complex reasoning. An increasing amount of work investigates how transformer language models (LMs) solve entity binding *without* state changes. However, there is limited understanding of how non-toy LMs address ET problems of realistic dif-

1. Introduction

Entity tracking (ET) is an ability essential for both humans and language models (LMs), recruited in a wide range of task contexts including playing chess (Karvonen, 2024), holding extended conversations across long contexts (Karttunen, 1969; Nieuwland & Van Berkum, 2006b; Kamp et al., 2011), or solving complex reasoning problems (Prakash et al., 2025). One benchmark that evaluates ET capacity

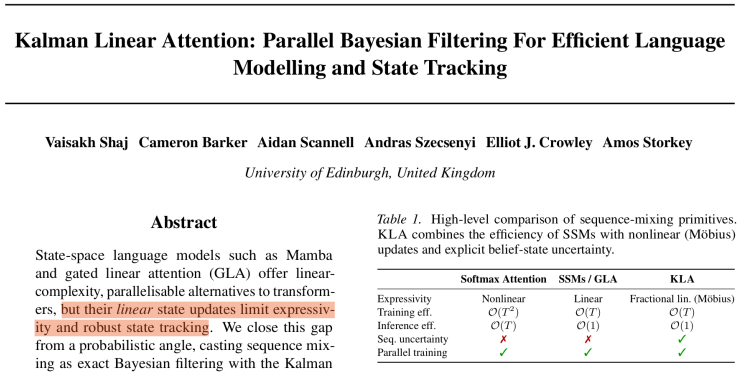
Representational Analysis of Binding in Language Models

Qin Dai[1], Benjamin Heinzerling[2,1], Kentaro Inui[3,1,2]
[1]Tohoku University [2]RIKEN AIP [3]MBZUAI
qin.dai.b8@tohoku.ac.jp, benjamin.heinzerling@riken.jp
kentaro.inui@mbzuai.ac.ae

Abstract

Entity tracking is essential for complex reasoning. To perform in-context entity tracking, language models (LMs) must bind an entity to its attribute (e.g., bind a container to its con-

tributes (Feng and Steinhardt, 2023). For exampl
given Sample 1 and 2, a model must bind the e
tities (e.g., "Box Z", "Box M", "Box H", "Alex
"John" and "Carl") to their corresponding attribute
(e.g., "coffee", "stone", "map", "bean", "pie" an



Kalman Linear Attention: Parallel Bayesian Filtering For Efficient Language Modelling and State Tracking

Vaisakh Shaj Cameron Barker Aidan Scannell Andras Szecsenyi Elliot J. Crowley Amos Storkey

University of Edinburgh, United Kingdom

Abstract

State-space language models such as Mamba and gated linear attention (GLA) offer linear-complexity, parallelisable alternatives to transformers, but their *linear* state updates limit expressivity and robust state tracking. We close this gap from a probabilistic angle, casting sequence mixing as exact Bayesian filtering with the Kalman

Table 1. High-level comparison of sequence-mixing primitives. KLA combines the efficiency of SSMs with nonlinear (Möbius) updates and explicit belief-state uncertainty.

	Softmax Attention	SSMs / GLA	KLA
Expressivity	Nonlinear	Linear	Fractional lin. (Möbius)
Training eff.	$\mathcal{O}(T^2)$	$\mathcal{O}(T)$	$\mathcal{O}(T)$
Inference eff.	$\mathcal{O}(T)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Seq. uncertainty	✗	✗	✓
Parallel training	✓	✓	✓

[1] Tang, Z., Zhao, Q., Franco, G., Wijaya, D., Mueller, A., Schuster, S., & Kim, N. (2026). *Do Language Models Track Entities Across State Changes?* arXiv. https://doi.org/10.48550/arXiv.2605.30233
[2] Dai, Q., Heinzerling, B., & Inui, K. (2024). *Representational Analysis of Binding in Large Language Models.* arXiv. https://doi.org/10.48550/arXiv.2409.05448
[3] Shaj, V., Barker, C., Scannell, A., Szecsenyi, A., Crowley, E. J., & Storkey, A. (2026). *Kalman Linear Attention: Parallel Bayesian Filtering for Efficient Language Modelling and State Tracking.* arXiv. https://doi.org/10.48550/arXiv.2602.10743

Multi-step Reasoning

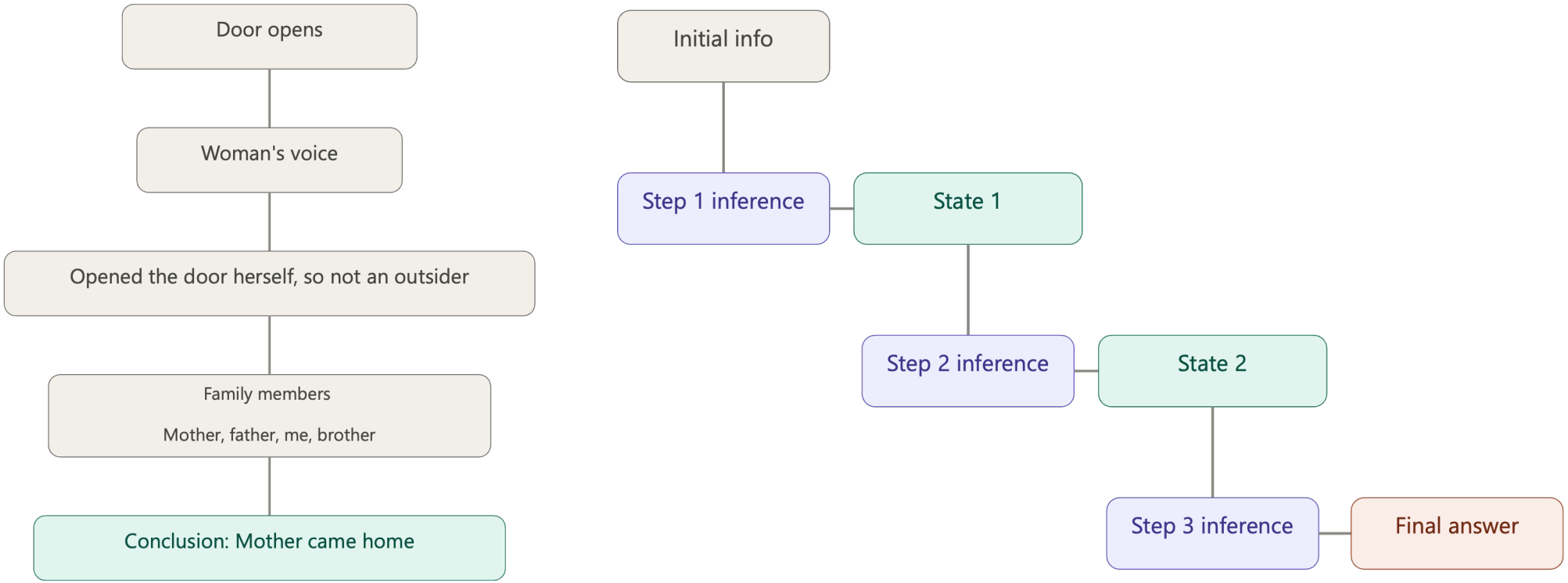


Fig. 1. Explicit chains expose each inference step; recurrent reasoning instead composes the same update operator in a latent state, without step-level supervision.

State Tracking and Recurrence

- **Limitations of Fixed-Depth Transformers**
  - (1) Transformers have a fixed computation depth **independent of input length**.
  - (2) A fixed depth impose **theoretical limits on sequence computation**.[1]

→ **Scaling the computation** required for **long-range state tracking** may be **difficult**.

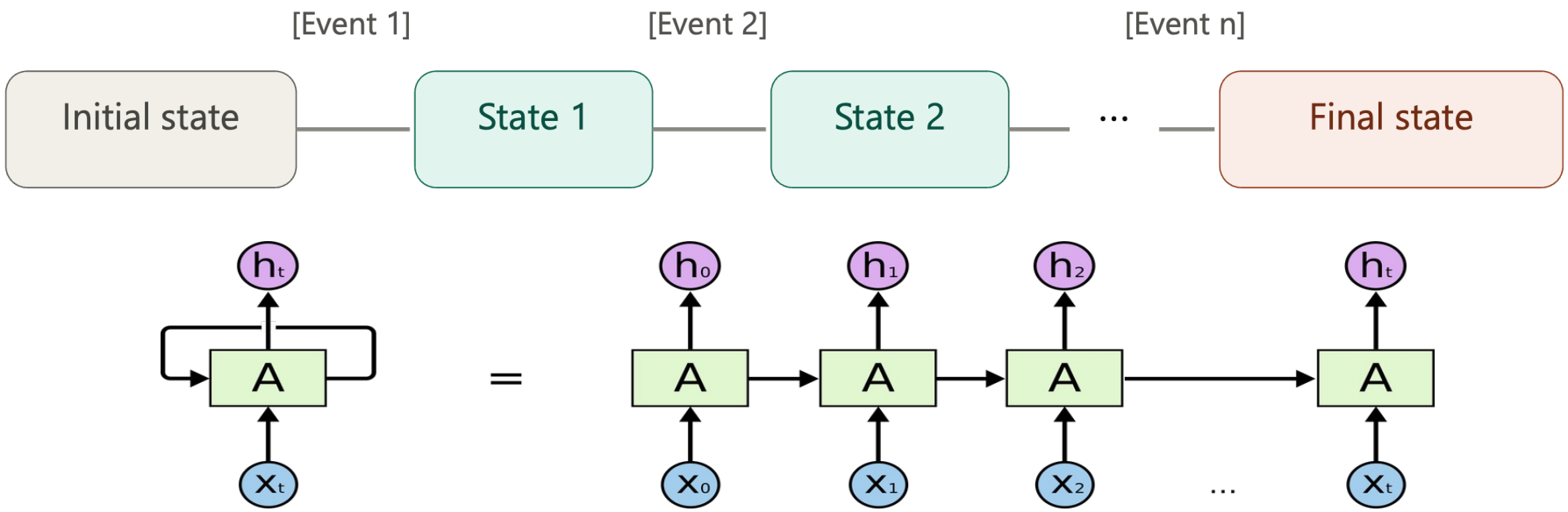


Fig. 2. Fixed-depth computation is constant, while the number of required state updates grows w/ the event sequence.

[1] The Parallelism Tradeoff: Limitations of Log-Precision Transformers (Merrill & Sabharwal, TACL 2023) https://arxiv.org/abs/2207.00729

State Tracking and Recurrence

- **Recurrent Transformer** extends fixed computation depth.[1]
- Increasing test-time **recurrence** scales latent **reasoning** compute.[2]

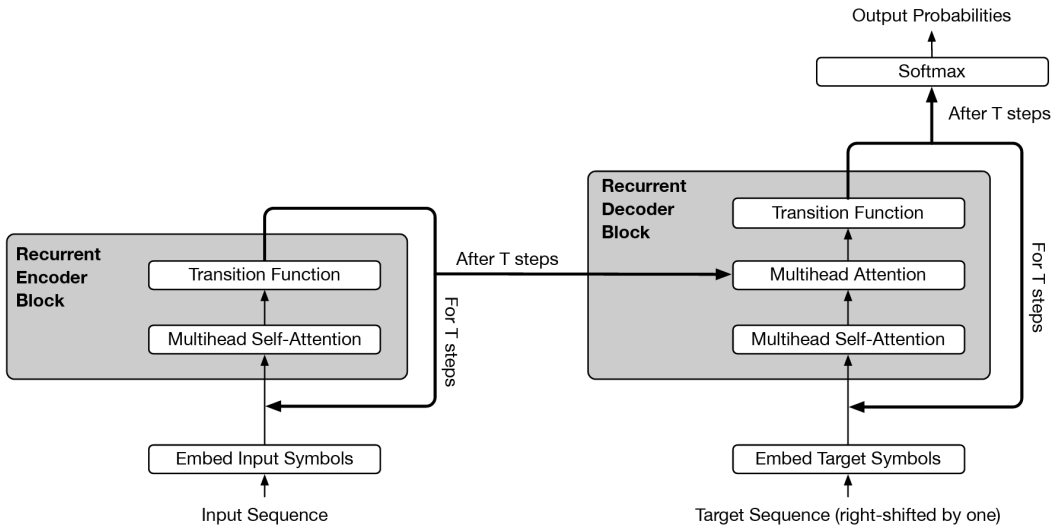


Fig. 3. Recurrent blocks of the Universal Transformer encoder and decoder

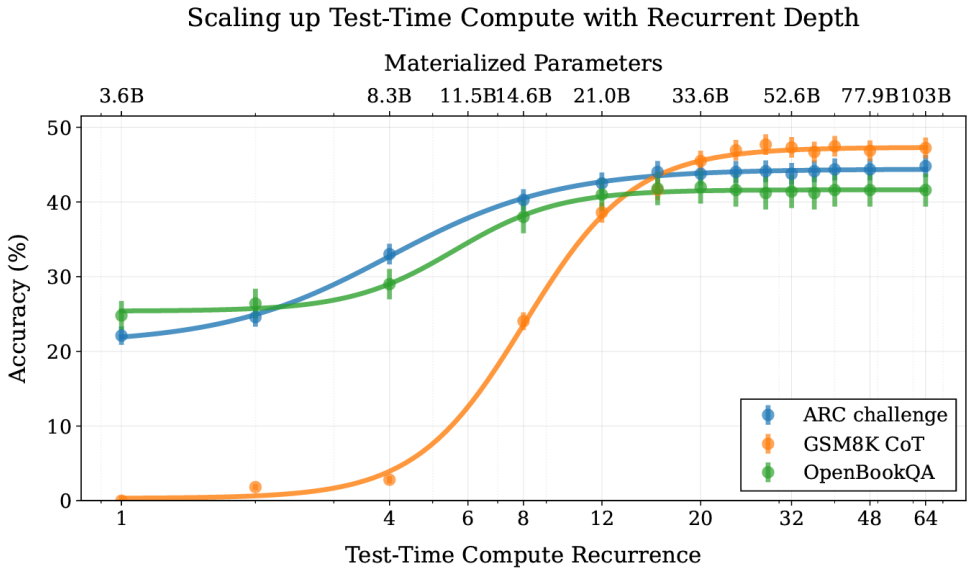


Fig. 4. Scaling up Test-Time Compute w/ Latent Reasoning by recurrent depth

[1] Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., & Kaiser, Ł. (2019). *Universal Transformers.* arXiv. https://doi.org/10.48550/arXiv.1807.03819
[2] Geiping, J., McLeish, S., Jain, N., Kirchenbauer, J., Singh, S., Bartoldson, B. R., Kailkhura, B., Bhatele, A., & Goldstein, T. (2025). *Scaling up Test-Time Compute with Latent Reasoning: A Recurrent Depth Approach.* arXiv. https://doi.org/10.48550/arXiv.2502.05171

Research Question

- RQ1: Does Fan-aligned recurrence generalize beyond fixed-depth baselines?
- RQ2: Which input representations and positional encodings support OOD extrapolation?
- RQ3: How do recurrent depth, width, and inference loops shift the generalization boundary?

Dataset

- 10K training examples; 500 examples per evaluation split; 5 entities × 5 colors
- Train/ID: 2–10 swaps | Boundary: 11–19 | OOD: 20–40 and 40–80

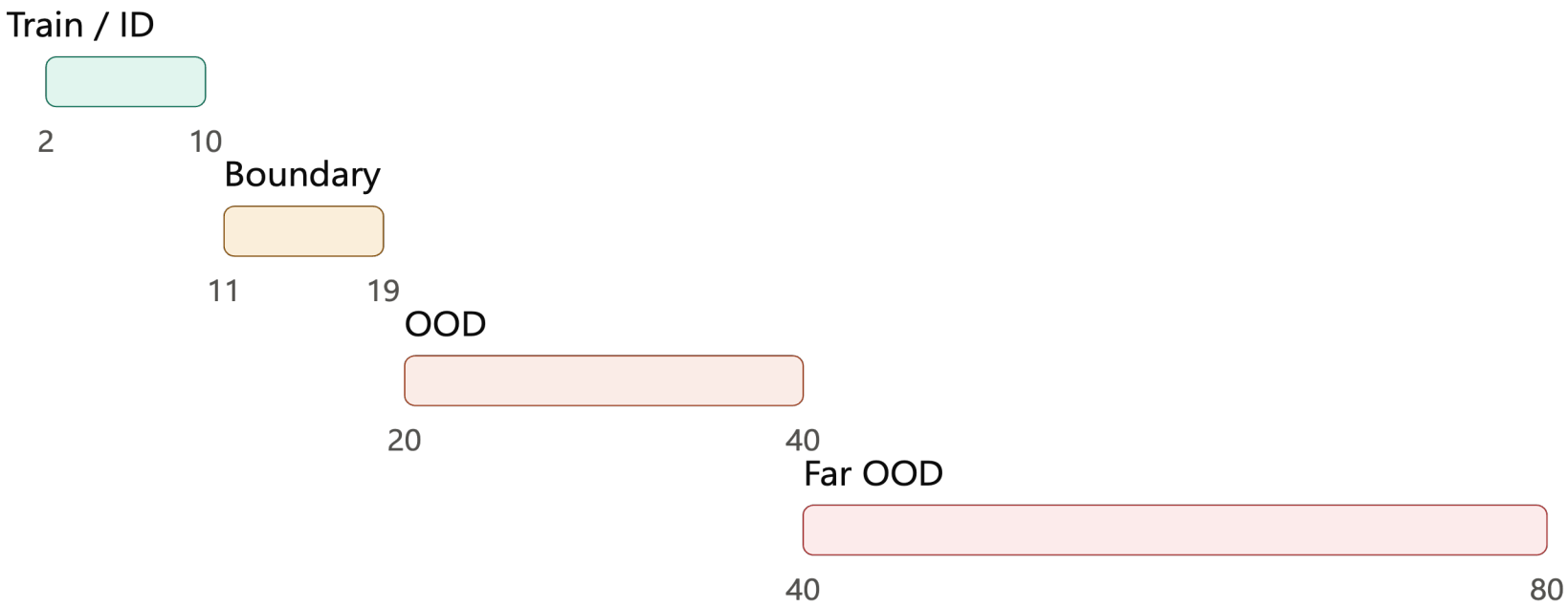


Fig. 5. Evaluation ranges isolate interpolation (2–10), the immediate boundary (11–19), and longer OOD rollouts (20–40, 40–80).

Model

- **Fixed-depth** Transformer vs. **shared-block recurrent** Transformer
- Same tokenizer, classifier, and final-state objective; length-matched compute uses K=N

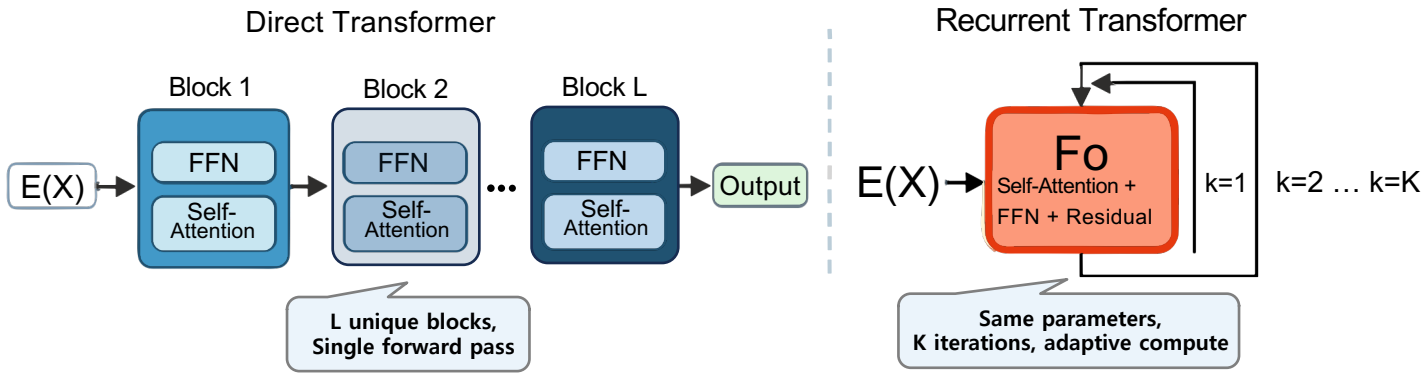


Fig. 6. Direct vs. Recurrent Transformer

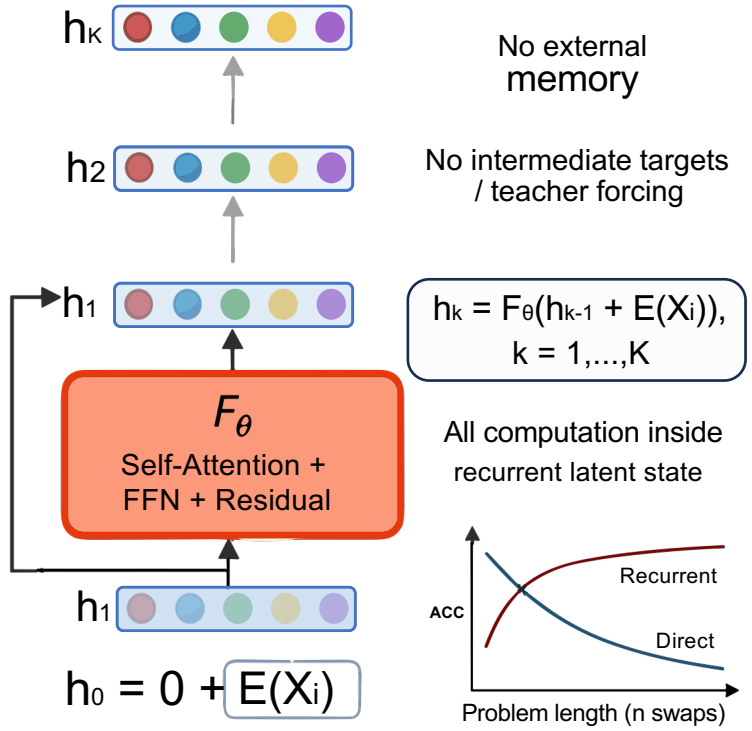


Fig. 7. Iterative Latent State Update (Recurrent Mechanism)

Training

- Online curriculum: maximum length grows 2→10 swaps; 100K optimizer steps total
- Atomic + NoPE: one swap per shared update, K=N; final-state CE only
- Seeds 0/1/2; stratified validation selects the lowest-loss checkpoint before test evaluation

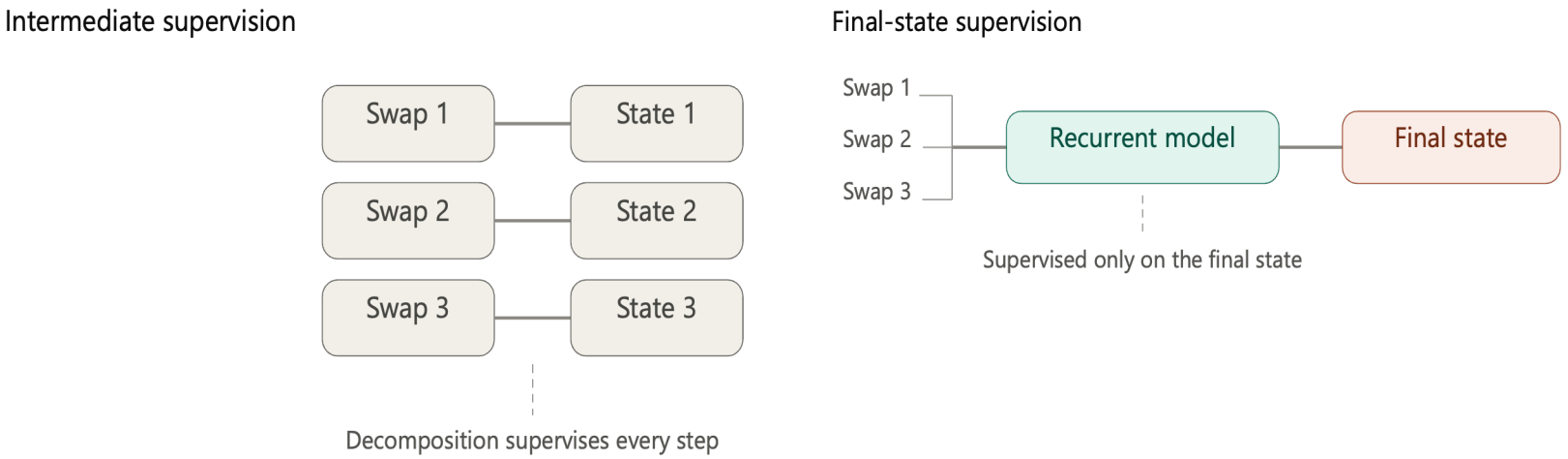


Fig. 8. We supervise only the final state; intermediate states are used for analysis, never for the training loss.

Importance of Input Representation

- Atomic + NoPE maintains 0.906 / 0.975 at the boundary, while other configurations degrade sharply.
- All configurations struggle at OOD 40–80, indicating a persistent limit in extreme length generalization.

(Exact Match / Slot ACC)

Token Type	Positional Encoding	Easy N=2	ID 2–10	Boundary 11–19	OOD 20–40	OOD 40–80
Natural	NoPE	0.000 / 0.205	0.000 / 0.201	0.000 / 0.199	0.000 / 0.199	0.000 / 0.190
Natural	Absolute Sinusoidal	1.000 / 1.000	1.000 / 1.000	0.003 / 0.278	0.000 / 0.206	0.000 / 0.198
Atomic	**NoPE**	**0.998 / 1.000**	**1.000 / 1.000**	**0.906 / 0.975**	**0.046 / 0.361**	**0.001 / 0.191**
Atomic	Absolute Sinusoidal	1.000 / 1.000	1.000 / 1.000	0.180 / 0.466	0.000 / 0.210	0.000 / 0.204

Table 1. Length generalization by input representation and positional encoding (3-seed mean where available).

Recurrence Outperforms Parameter-Matched Baselines

- At 0.24M parameters, recurrence raises Boundary performance from 0.001 / 0.209 to 0.906 / 0.975.
- The 3.31M recurrent model also exceeds the 9.63M oversized baseline, isolating recurrence—not parameter count—as the key gain.
- Exact Match remains near zero at OOD 40–80, so extreme extrapolation is still unresolved.

(Exact Match / Slot ACC)

Model	Params	Matching Criterion	ID 2–10	Boundary 11–19	OOD 20–40	OOD 40–80
Basic d128 L1	0.24M	Parameter-matched	0.099 / 0.428	0.001 / 0.209	0.000 / 0.202	0.001 / 0.204
Basic d128 L7	1.43M	FLOPs-matched	1.000 / 1.000	0.281 / 0.539	0.000 / 0.202	0.000 / 0.201
Basic d128 L10	2.02M	Max-train-recurrence-matched	1.000 / 1.000	0.332 / 0.584	0.000 / 0.203	0.000 / 0.197
Basic d256 L12	9.63M	Strong oversized baseline	1.000 / 1.000	0.381 / 0.639	0.000 / 0.207	0.000 / 0.198
Recurrent d128 L1	**0.24M**	Main recurrent model	**1.000 / 1.000**	**0.906 / 0.975**	**0.046 / 0.361**	0.001 / 0.191
Recurrent d256 L4	**3.31M**	Scaled recurrent model	1.000 / 1.000	**0.994 / 0.999**	**0.351 / 0.661**	**0.002 / 0.220**

Table 2. Fixed-depth baselines versus recurrent models under parameter-, compute-, and oversized controls.

Latent trajectories separate outcomes

- Loop color tracks state evolution; successful and failed paths separate (824 vs. 76 examples).

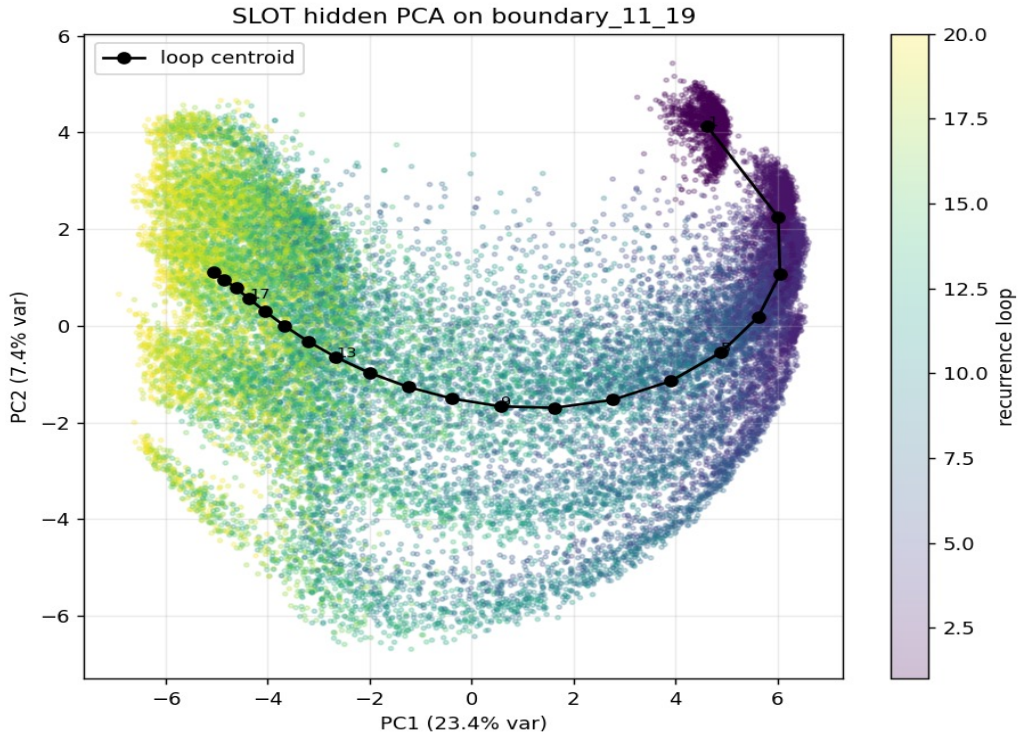


Fig. 9. PCA of normalized SLOT states across recurrence loops (Boundary 11–19).

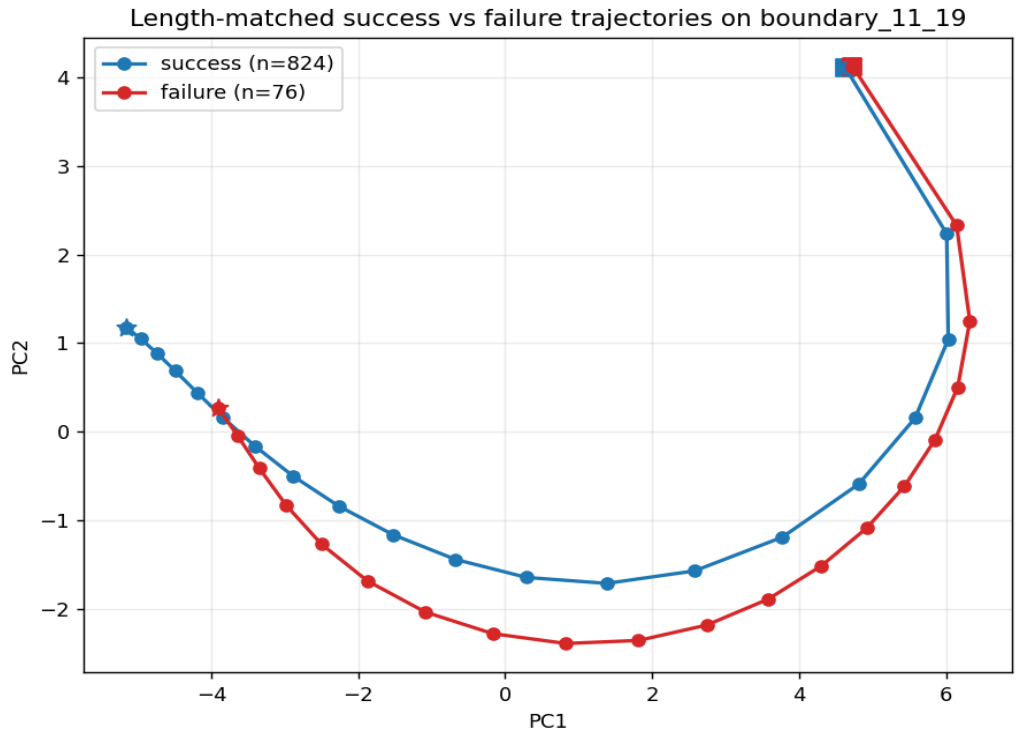


Fig. 10. Mean PCA trajectories grouped by final Exact Match; predictions are read at K=N.

Performance

- Depth and moderate periodic encoding improve OOD 20–40 more consistently than width alone.
  - Values = Exact Match / Slot Accuracy. "Best" = max over a fixed-K sweep on that split (diagnostic upper bound, not validation-selected).

(Exact Match / Slot ACC)

Condition	Params	Boundary 11–19	OOD 20–40	OOD 20–40 (Best)	OOD 40–80
NoPE d128 L1	0.24M	0.906 / 0.975	0.046 / 0.361	0.092 / 0.400	0.001 / 0.191
NoPE d128 L2	0.44M	0.961 / 0.990	0.123 / 0.464	0.189 / 0.518	0.000 / 0.208
NoPE d128 L4	**0.84M**	**0.982 / 0.995**	**0.231 / 0.567**	**0.250 / 0.571**	0.001 / 0.212
NoPE d128 L6	1.23M	0.975 / 0.993	0.193 / 0.525	0.208 / 0.529	0.000 / 0.203
Periodic P=1 d128 L6	1.23M	0.980 / 0.994	0.199 / 0.540	0.237 / 0.553	0.001 / 0.204
Periodic P=2 d128 L6	**1.23M**	**0.990 / 0.997**	**0.273 / 0.620**	**0.299 / 0.628**	**0.001 / 0.216**
Periodic P=5 d128 L6	1.23M	0.495 / 0.772	0.029 / 0.310	0.043 / 0.323	0.000 / 0.202
NoPE d256 L1	0.94M	0.964 / 0.990	0.121 / 0.450	0.150 / 0.479	0.001 / 0.209
NoPE d256 L2	1.73M	0.987 / 0.997	0.243 / 0.582	0.302 / 0.615	0.000 / 0.206
NoPE d256 L4	**3.31M**	**0.994 / 0.999**	**0.351 / 0.661**	**0.375 / 0.672**	**0.002 / 0.220**
NoPE d512 L1	3.71M	0.980 / 0.995	0.151 / 0.477	0.195 / 0.494	0.000 / 0.204

Table 3. Seed-0 scaling ablations; "Best" reports the strongest fixed-K diagnostic result for OOD 20–40.

Scaling shifts the OOD boundary

- Width (L1, seed 0, K=N): d128→d512 raises OOD 20–40 Exact Match 4.2%→18.2%
- Depth (d256, seed 0, K=N): L1→L4 raises 6.0%→35.6%
- OOD 40–80 ≈0%: scaling shifts the limit; it does not solve extreme extrapolation

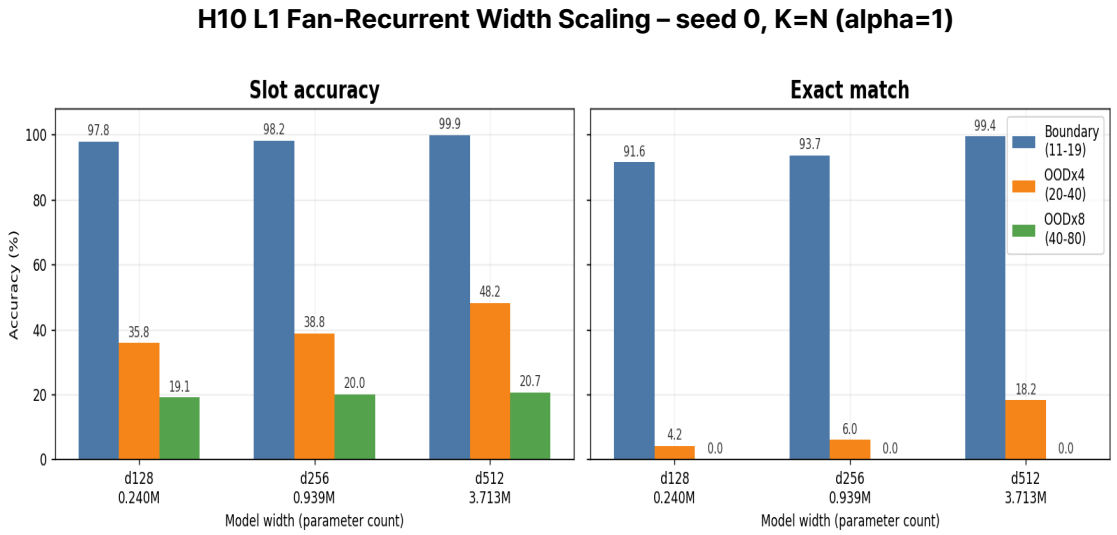


Fig. 11. Width scaling of the L1 Fan-recurrent model (seed 0, K=N).

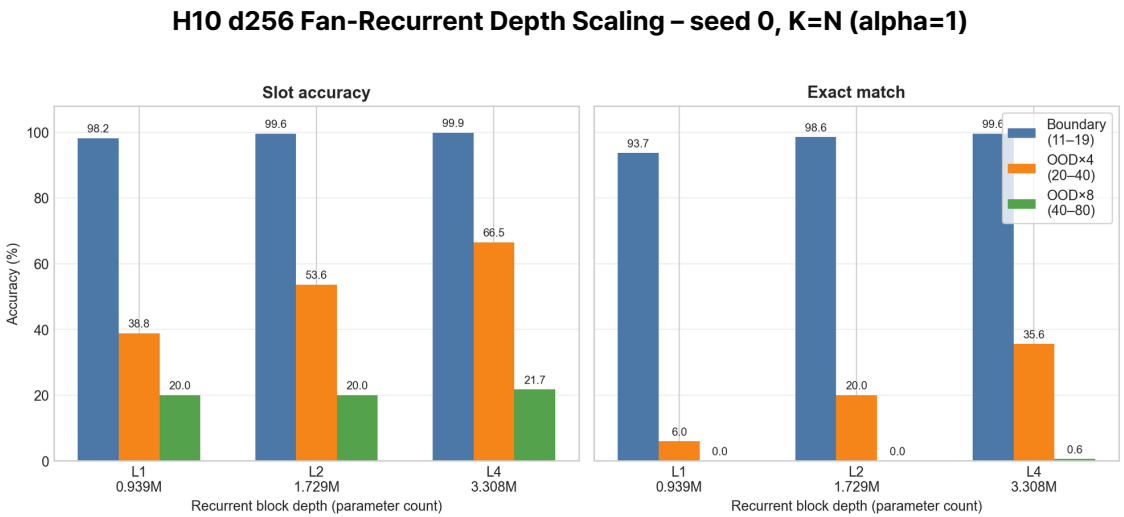


Fig. 12. Recurrent-depth scaling at d256 (seed 0, K=N).

Conclusion

- Fan-aligned recurrence outperforms matched fixed-depth baselines beyond the training range.
- **Length-invariant atomic tokens and NoPE are critical; absolute position cues cause early collapse.**
- Scaling pushes the OOD boundary, but 40–80-swap Exact Match remains near zero.
- More recurrence is not always better: over-computation can produce state drift.

[1] Fan, Y., Du, Y., Ramchandran, K., & Lee, K. (2025). Looped Transformers for Length Generalization. arXiv. https://doi.org/10.48550/arXiv.2409.15647

Thank you