

Improving Signal Propagation in Looped Transformers

Shlomo Libo Feigin

25.06.2026

based on

Fixed-Point Reasoners: Stable and Adaptive Deep Looped Transformers

Movahedi, Milovanović, Feigin, Theus, Hofmann, Boeva, Rusch, Orvieto • arXiv:2606.18206



MAX PLANCK INSTITUTE
FOR INTELLIGENT SYSTEMS



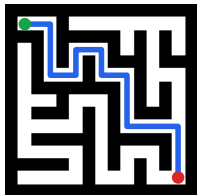
ETH zürich

Outline

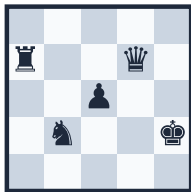
- Tasks Transformers *cannot* learn
- The standard fix: Chain-of-Thought
- The alternative: **Looped Transformers**
- **Signal propagation** in Looped Transformers
- Recent work: **Improving signal propagation**

From vanilla Transformers to Looped Transformers

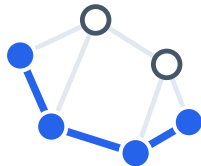
Tasks a (vanilla) Transformer cannot solve



shortest path
in a maze

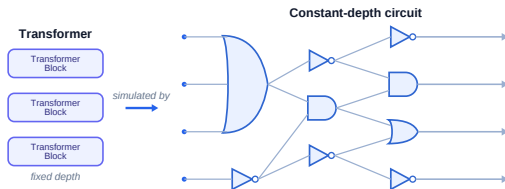


state tracking



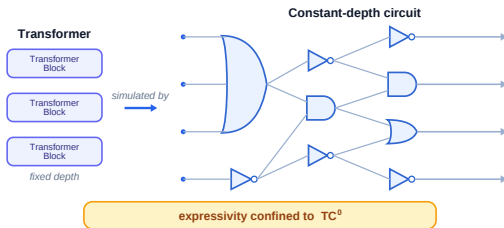
graph reachability

The reason is a complexity theoretic limit



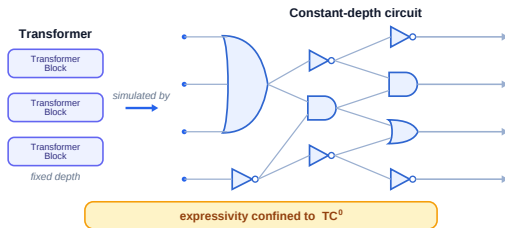
- A **fixed-depth** Transformer can be simulated by a **constant-depth** Boolean circuit. [Merrill & Sabharwal]

The reason is a complexity theoretic limit



- A **fixed-depth** Transformer can be simulated by a **constant-depth** Boolean circuit. [Merrill & Sabharwal]
- TC^0 = what boolean circuits can compute with **constant depth**, polynomial size, unlimited fan-in and majority gates.

The reason is a complexity theoretic limit



- A **fixed-depth** Transformer can be simulated by a **constant-depth** Boolean circuit. [Merrill & Sabharwal]
- TC^0 = what boolean circuits can compute with **constant depth**, polynomial size, unlimited fan-in and majority gates.
- These tasks lie **outside** TC^0 .

Standard fix: Chain-of-Thought

Can you explain this argument to me?

Unpacked mathematical argument connecting variance bounds to gradient concentration ∨

⌚ The user is asking about a specific argument from a paper on gradient-based deep learning, where the authors use Theorem 1 to show that variance decreases exponentially with dimension, and then apply Chebyshev's inequality to argue that gradients become highly concentrated around a fixed point regardless of the specific input.

Now I'm working through the mathematical setup to explain this properly. The core idea involves learning random parity functions from a hypothesis class of size 2^d , and the key quantity is the variance of gradients across different target functions—measuring how much the gradient changes depending on which function we're trying to learn. The authors then use this variance bound to show that the gradient becomes increasingly concentrated as dimension grows. This bound is exponentially small in the dimension. The key insight is that the average gradient $\bar{g}(w) = \mathbb{E}_H \nabla F_H(w)$ is a fixed vector independent of which specific target we're looking at, and the variance tells us how tightly the individual gradients cluster around this fixed point. Applying Markov's inequality to bound the probability that the gradient deviates from this average by more than some threshold δ , I get that this probability decays exponentially in the dimension d . This means with overwhelming probability over random choices of the target, the actual gradient stays close to the fixed average.

Standard fix: Chain-of-Thought

Can you explain this argument to me?

Unpacked mathematical argument connecting variance bounds to gradient concentration ∨

🕒 The user is asking about a specific argument from a paper on gradient-based deep learning, where the authors use Theorem 1 to show that variance decreases exponentially with dimension, and then apply Chebyshev's inequality to argue that gradients become highly concentrated around a fixed point regardless of the specific input.

Now I'm working through the mathematical setup to explain this properly. The core idea involves learning random parity functions from a hypothesis class of size 2^d , and the key quantity is the variance of gradients across different target functions—measuring how much the gradient changes depending on which function we're trying to learn. The authors then use this variance bound to show that the gradient becomes increasingly concentrated as dimension grows. This bound is exponentially small in the dimension. The key insight is that the average gradient $\bar{g}(w) = \mathbb{E}_H \nabla F_H(w)$ is a fixed vector independent of which specific target we're looking at, and the variance tells us how tightly the individual gradients cluster around this fixed point. Applying Markov's inequality to bound the probability that the gradient deviates from this average by more than some threshold δ , I get that this probability decays exponentially in the dimension d . This means with overwhelming probability over random choices of the target, the actual gradient stays close to the fixed average.

- Chain-of-Thought adds depth.

Standard fix: Chain-of-Thought

Can you explain this argument to me?

Unpacked mathematical argument connecting variance bounds to gradient concentration ▾

🕒 The user is asking about a specific argument from a paper on gradient-based deep learning, where the authors use Theorem 1 to show that variance decreases exponentially with dimension, and then apply Chebyshev's inequality to argue that gradients become highly concentrated around a fixed point regardless of the specific input.

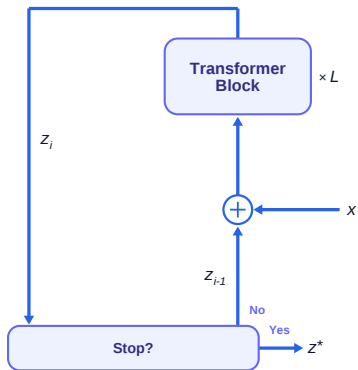
Now I'm working through the mathematical setup to explain this properly. The core idea involves learning random parity functions from a hypothesis class of size 2^d , and the key quantity is the variance of gradients across different target functions—measuring how much the gradient changes depending on which function we're trying to learn. The authors then use this variance bound to show that the gradient becomes increasingly concentrated as dimension grows. This bound is exponentially small in the dimension. The key insight is that the average gradient $\bar{g}(w) = \mathbb{E}_H \nabla F_H(w)$ is a fixed vector independent of which specific target we're looking at, and the variance tells us how tightly the individual gradients cluster around this fixed point. Applying Markov's inequality to bound the probability that the gradient deviates from this average by more than some threshold δ , I get that this probability decays exponentially in the dimension d . This means with overwhelming probability over random choices of the target, the actual gradient stays close to the fixed average.

- Chain-of-Thought adds depth.
- But training requires **hand-crafted** reasoning traces — not learned end-to-end.

The alternative: Looped Transformers

- Loop a transformer until done.

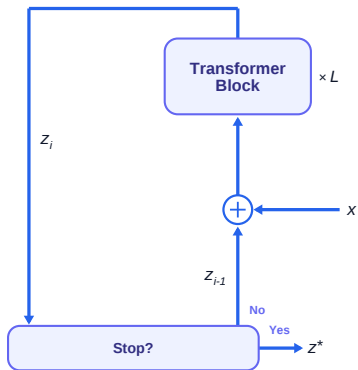
$$\mathbf{z}_i = f_{\theta}(\mathbf{z}_{i-1}; \mathbf{x})$$



The alternative: Looped Transformers

- Loop a transformer until done.
- Learns to reason **end-to-end**.

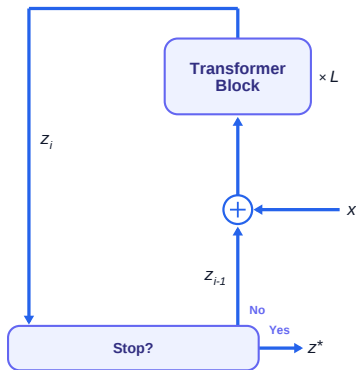
$$\mathbf{z}_i = f_{\theta}(\mathbf{z}_{i-1}; \mathbf{x})$$



The alternative: Looped Transformers

- Loop a transformer until done.
- Learns to reason **end-to-end**.
- Compute **scales with number of loops**:

$$\mathbf{z}_i = f_{\theta}(\mathbf{z}_{i-1}; \mathbf{x})$$

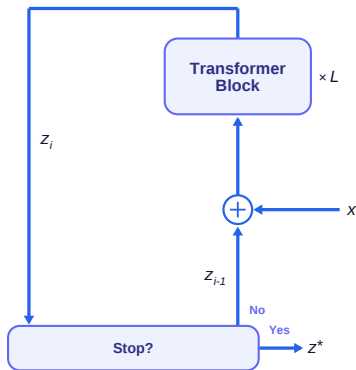


The alternative: Looped Transformers

- Loop a transformer until done.
- Learns to reason **end-to-end**.
- Compute **scales with number of loops**:

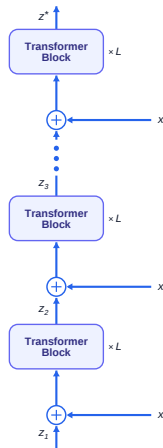
$$\mathbf{z}_i = f_{\theta}(\mathbf{z}_{i-1}; \mathbf{x})$$

- We use reaching a **fixed point** as a halting criterion.



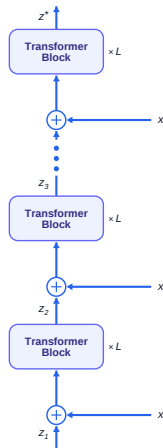
Looped Transformers are deep Transformers

- Unrolling the loop yields a **very deep** network.



Looped Transformers are deep Transformers

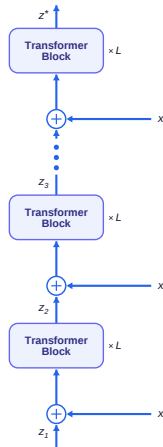
- Unrolling the loop yields a **very deep** network.
- Training deep models is *hard*.



Looped Transformers are deep Transformers

- Unrolling the loop yields a **very deep** network.
- Training deep models is *hard*.

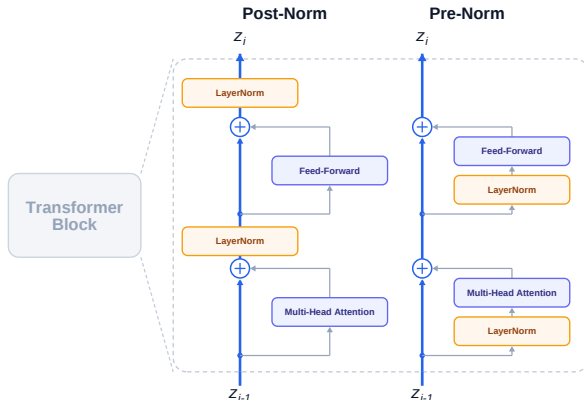
⇒ **Signal propagation** becomes a bottleneck.



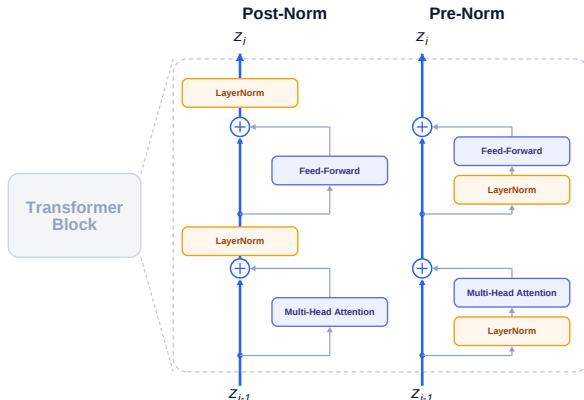


Signal propagation in Looped Transformers

Two ways to place the normalization

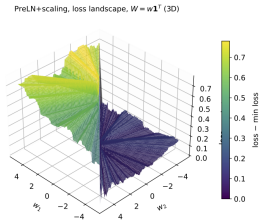
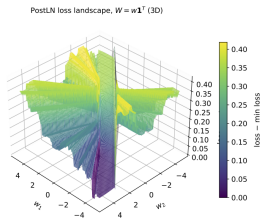
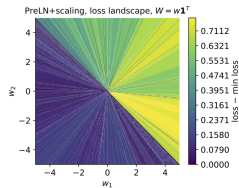
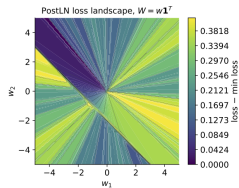


Two ways to place the normalization



Looped models typically use **post-norm**; deep (non-looped) networks prefer **pre-norm**.

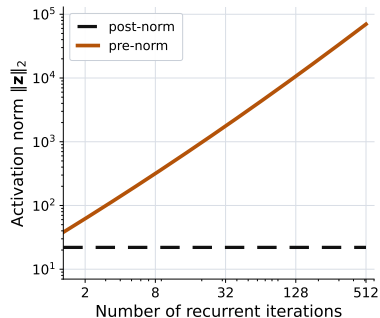
Pre-norm makes transformers more trainable



Post-norm (left): narrow loss ridges

Pre-norm (right): better conditioned.

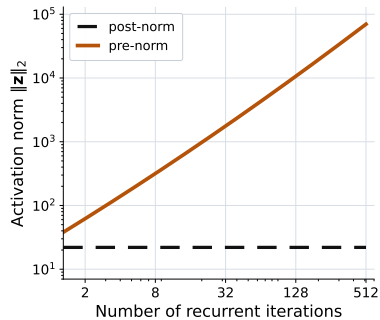
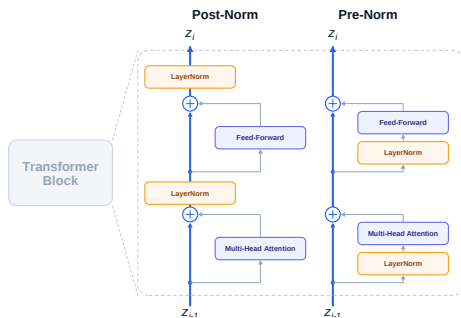
...but pre-norm causes activations to explode



activation norm at initialization

Post-norm stays bounded while **pre-norm** causes activations to **explode**.

...but pre-norm causes activations to explode



activation norm at initialization

Post-norm stays bounded while **pre-norm** causes activations to **explode**.

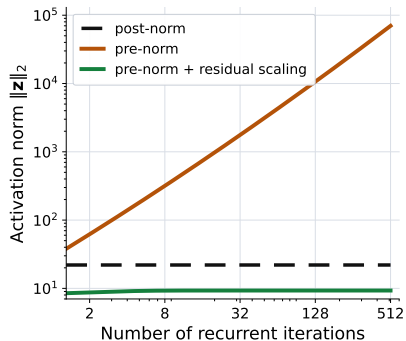
Can we switch to **pre-norm** while keeping the activations **bounded**?

Can we switch to **pre-norm** while keeping the activations **bounded**?

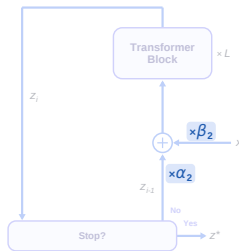
Yes — via residual scaling.

Can we switch to **pre-norm** while keeping the activations **bounded**?

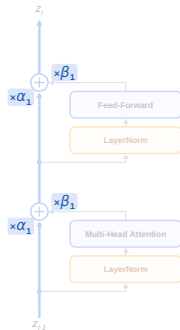
Yes — via residual scaling.



Recovering boundedness via residual scaling



across iterations (α_2, β_2)



within a block (α_1, β_1)

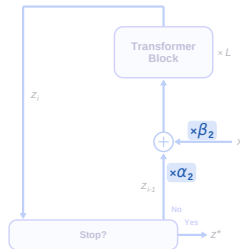
Recovering boundedness via residual scaling

Theorem 1 (boundedness).

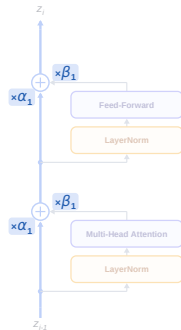
With $0 \leq \alpha_1, \alpha_2 < 1$ and the coupling $\beta_2 = 1 - \alpha_2 \alpha_1^{2L}$,

$\beta_1 = \frac{\beta_2(1-\alpha_1)}{1-\alpha_1^{2L}}$, the iterates stay **bounded**:

$$\|\mathbf{z}_\infty^0\| \leq \|\mathbf{x}\| + \alpha_2 C_f.$$



across iterations (α_2, β_2)



within a block (α_1, β_1)

Putting it together: Fixed-Point Reasoning Model (FPRM)

The benchmark tasks

The benchmark tasks

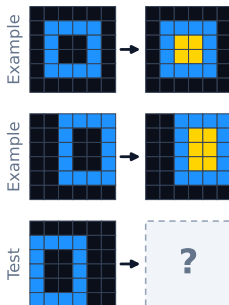
						1	
4							
	2						
			5		4		7
		8			3		
		1	9				
3			4		2		
	5		1				
			8		6		

Sudoku-Extreme

The benchmark tasks

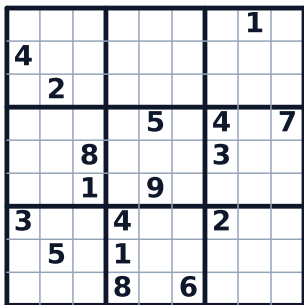
4						1
	2					
			5		4	7
		8			3	
		1		9		
3			4			2
	5		1			
			8		6	

Sudoku-Extreme

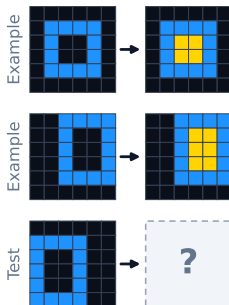


ARC-AGI

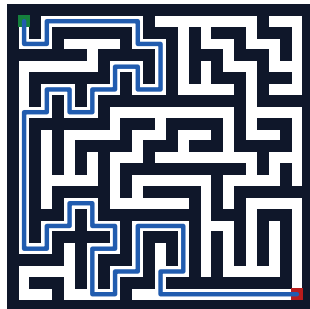
The benchmark tasks



Sudoku-Extreme



ARC-AGI



Maze-Hard

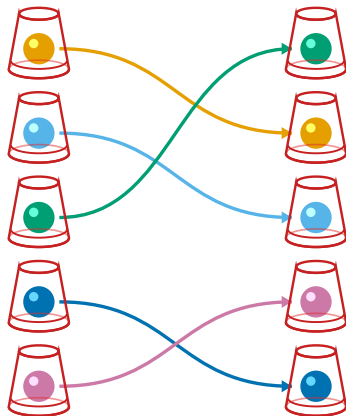
FPRM performance

Model	# par.	No Hier.	Sudoku-Ext.	Maze-Hard	ARC-1	ARC-2
HRM	27M	$\times$	55.0	74.5	40.3	5.0
Attractor	27M	$\times$	91.4	93.1	—	—
URM	14M	$\times$	77.6	—	53.8	16.0
TRM	7M	$\times$	74.7	85.3	44.6	<u>7.8</u>
EqR	7M	$\times$	93.0	—	—	—
FPRM	7M	$\checkmark$	<u>94.2</u>	<u>87.0</u>	<u>47.5</u>	6.2

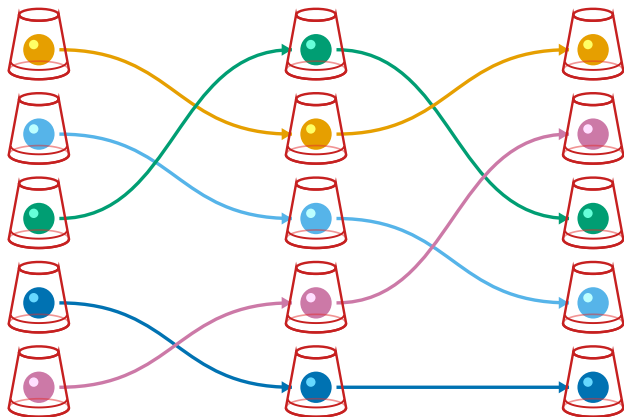
The S_5 benchmark: composing permutations



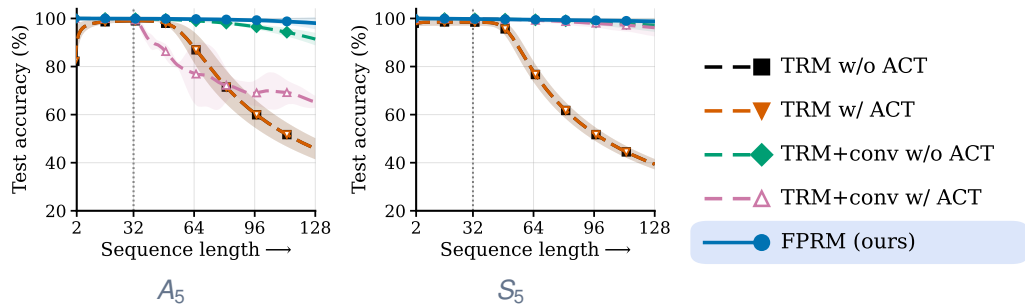
The S_5 benchmark: composing permutations



The S_5 benchmark: composing permutations

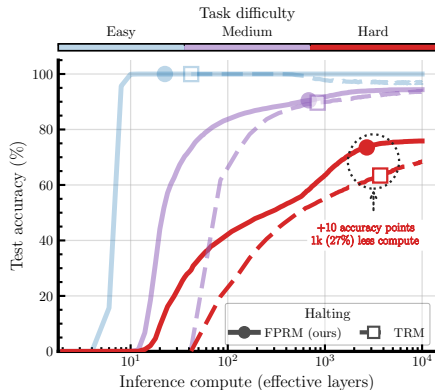


FPRM can learn to do state tracking



On state tracking, FPRM holds $\sim 100\%$ accuracy far beyond the training length (32, dotted).

FPRM adapts compute to instance difficulty



Sudoku-Extreme accuracy vs. compute across difficulty.

Summary

- Looped Transformers are, in part, **very deep** Transformers — so **signal propagation** matters.

Summary

- Looped Transformers are, in part, **very deep** Transformers — so **signal propagation** matters.
- **Post-norm**: bounded, but propagates signal poorly.
Pre-norm: propagates well, but is unbounded.

Summary

- Looped Transformers are, in part, **very deep** Transformers — so **signal propagation** matters.
- **Post-norm**: bounded, but propagates signal poorly.
Pre-norm: propagates well, but is unbounded.
- **Residual scaling** makes pre-norm **bounded** $\Rightarrow$ trainable at depth.

References

1. S. Movahedi, V. Milovanović, S. L. Feigin, A. Theus, T. Hofmann, V. Boeva, T. K. Rusch, A. Orvieto. *Fixed-Point Reasoners: Stable and Adaptive Deep Looped Transformers*. arXiv:2606.18206, 2026.
2. W. Merrill, A. Sabharwal, N. A. Smith. *Saturated Transformers are Constant-Depth Threshold Circuits*. 2022.
3. G. Wang et al. *Hierarchical Reasoning Model*. 2025.
4. A. Jolicoeur-Martineau. *Less is More: Recursive Reasoning with Tiny Networks*. 2025.
5. F. Chollet, M. Knoop, G. Kamradt, B. Landers. *ARC Prize 2024: Technical Report*. 2024.

Thanks to my collaborators



Sajad Movahedi



Vera Milovanović



Alexander Theus



Thomas Hofmann



Valentina Boeva



T. Konstantin
Rusch



Antonio Orvieto

Thank you!



Read the paper | [arXiv:2606.18206](https://arxiv.org/abs/2606.18206)