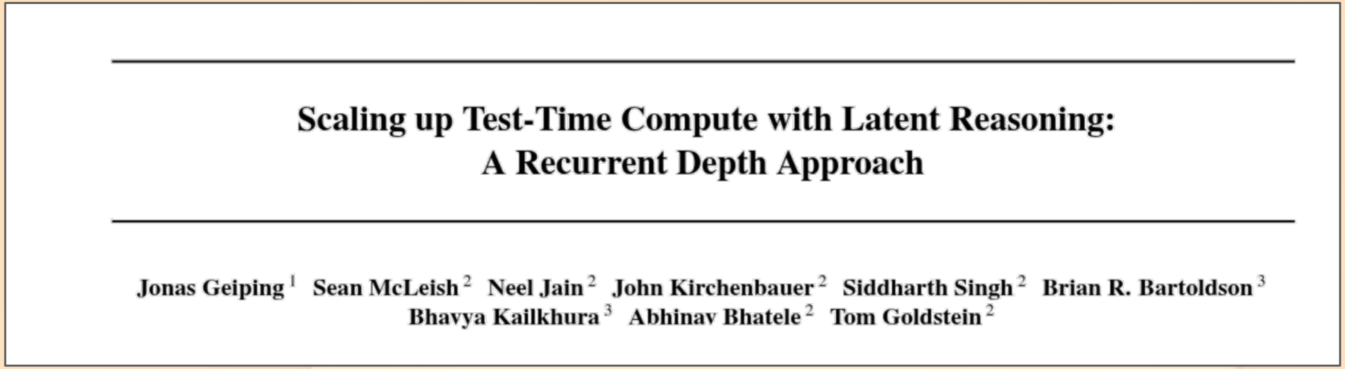
What's so interesting about models with recurrent depth?

Jonas Geiping

ellis
INSTITUTE
TÜBINGEN

MAX PLANCK INSTITUTE
FOR INTELLIGENT SYSTEMS



Scaling up Test-Time Compute with Latent Reasoning: A Recurrent Depth Approach

Jonas Geiping [1] Sean McLeish [2] Neel Jain [2] John Kirchenbauer [2] Siddharth Singh [2] Brian R. Bartoldson [3]
Bhavya Kailkhura [3] Abhinav Bhatele [2] Tom Goldstein [2]

Paper: https://www.arxiv.org/abs/2502.05171

Code: github.com/seal-rg/recurrentpretraining

Model: huggingface.co/tomg-group-umd/huginn-0125

Data (still uploading): huggingface.co/datasets/tomg-group-umd/huginn-dataset

Key Concepts

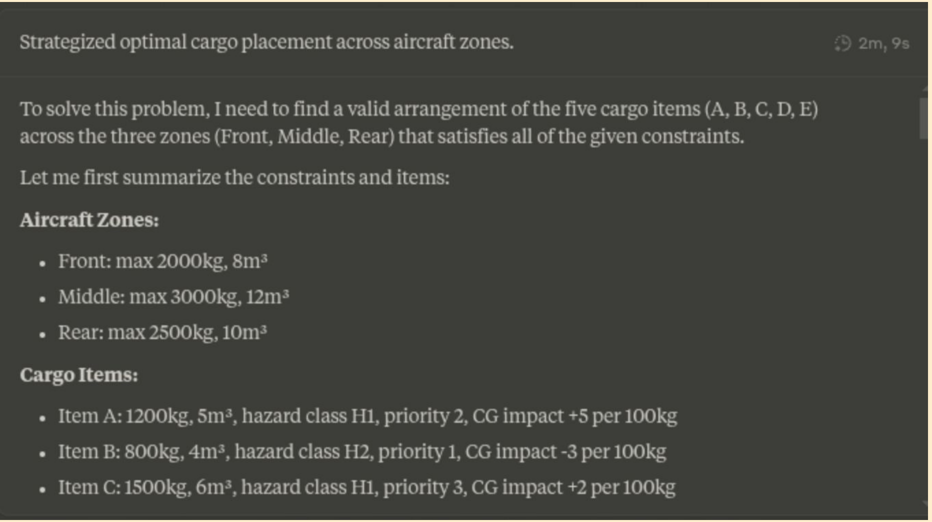
What is Test-Time Compute?

Improving the performance of a machine learning model by using more computation at test-time/ for inference.

What is "Reasoning"

Learning to pattern match abstract concepts such as meta-strategies, logic, abstraction and plans, instead of just memorizing facts.

What is *Verbalized* Reasoning



Strategized optimal cargo placement across aircraft zones.

To solve this problem, I need to find a valid arrangement of the five cargo items (A, B, C, D, E) across the three zones (Front, Middle, Rear) that satisfies all of the given constraints.

Let me first summarize the constraints and items:

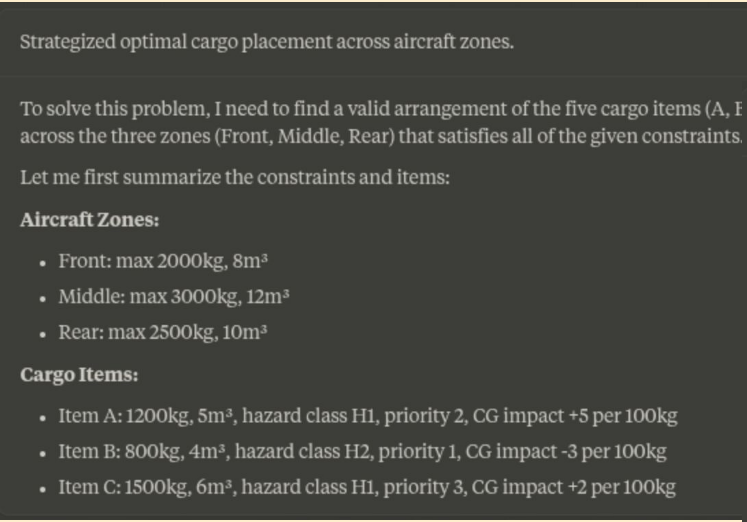
Aircraft Zones:

- Front: max 2000kg, 8m³
- Middle: max 3000kg, 12m³
- Rear: max 2500kg, 10m³

Cargo Items:

- Item A: 1200kg, 5m³, hazard class H1, priority 2, CG impact +5 per 100kg
- Item B: 800kg, 4m³, hazard class H2, priority 1, CG impact -3 per 100kg
- Item C: 1500kg, 6m³, hazard class H1, priority 3, CG impact +2 per 100kg

Verbalized Reasoning



Strategized optimal cargo placement across aircraft zones.

2m, 9s

To solve this problem, I need to find a valid arrangement of the five cargo items (A, across the three zones (Front, Middle, Rear) that satisfies all of the given constraints.

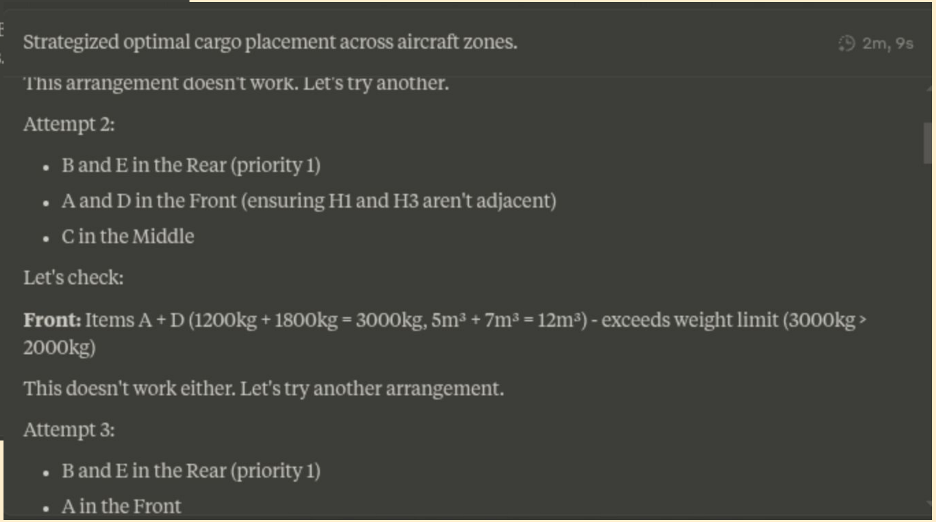
Let me first summarize the constraints and items:

Aircraft Zones:

- Front: max 2000kg, 8m^3
- Middle: max 3000kg, 12m^3
- Rear: max 2500kg, 10m^3

Cargo Items:

- Item A: 1200kg, 5m^3, hazard class H1, priority 2, CG impact +5 per 100kg
- Item B: 800kg, 4m^3, hazard class H2, priority 1, CG impact -3 per 100kg
- Item C: 1500kg, 6m^3, hazard class H1, priority 3, CG impact +2 per 100kg



Strategized optimal cargo placement across aircraft zones.

2m, 9s

This arrangement doesn't work. Let's try another.

Attempt 2:

- B and E in the Rear (priority 1)
- A and D in the Front (ensuring H1 and H3 aren't adjacent)
- C in the Middle

Let's check:

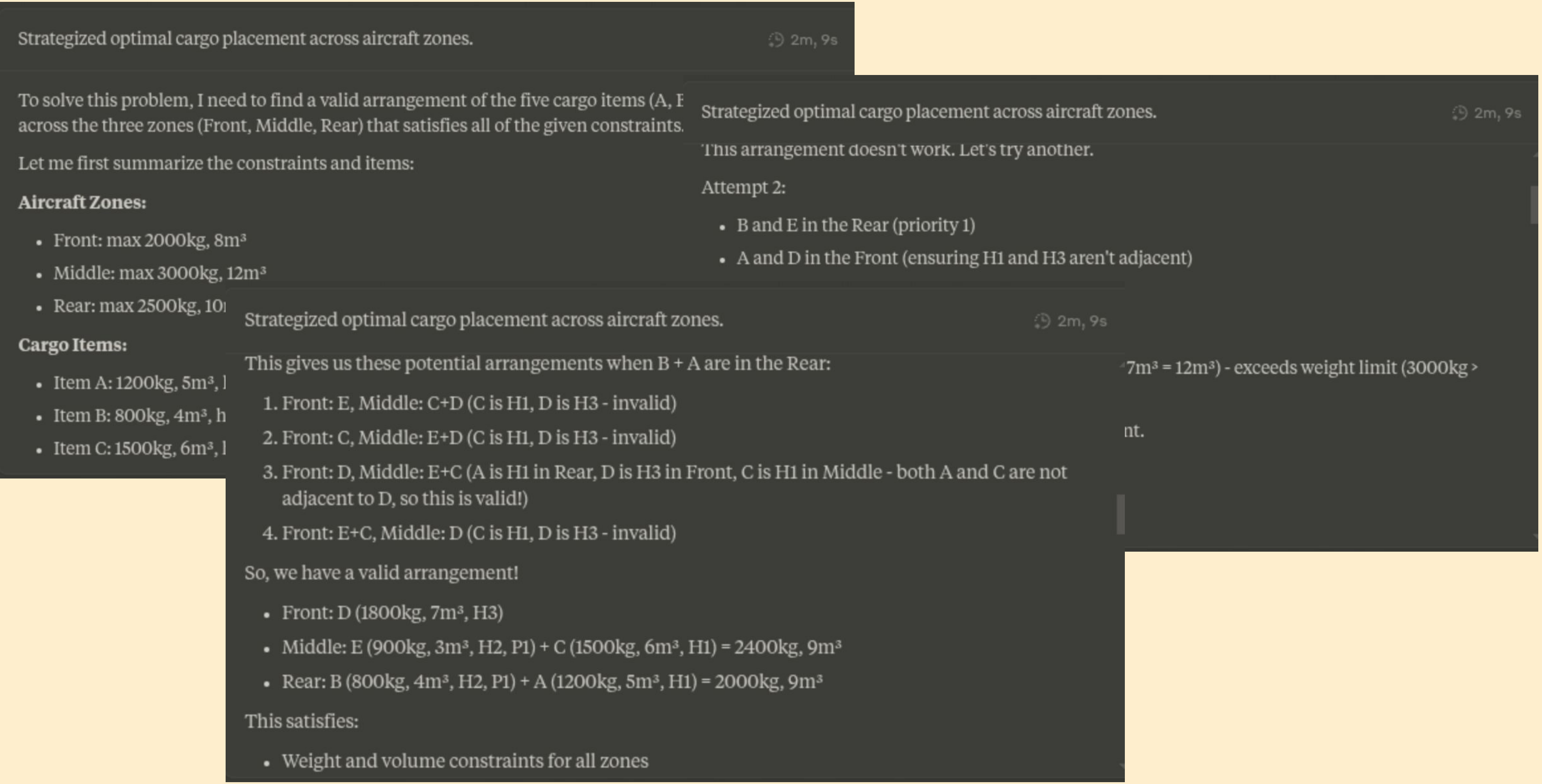
Front: Items A + D (1200kg + 1800kg = 3000kg, 5m^3 + 7m^3 = 12m^3) - exceeds weight limit (3000kg > 2000kg)

This doesn't work either. Let's try another arrangement.

Attempt 3:

- B and E in the Rear (priority 1)
- A in the Front

Verbalized Reasoning



Strategized optimal cargo placement across aircraft zones.

To solve this problem, I need to find a valid arrangement of the five cargo items (A,
across the three zones (Front, Middle, Rear) that satisfies all of the given constraints.

Let me first summarize the constraints and items:

Aircraft Zones:

- Front: max 2000kg, 8m^3
- Middle: max 3000kg, 12m^3
- Rear: max 2500kg, 10

Cargo Items:

- Item A: 1200kg, 5m^3,
- Item B: 800kg, 4m^3, h
- Item C: 1500kg, 6m^3,

Strategized optimal cargo placement across aircraft zones.

This arrangement doesn't work. Let's try another.

Attempt 2:

- B and E in the Rear (priority 1)
- A and D in the Front (ensuring H1 and H3 aren't adjacent)

7m^3 = 12m^3) - exceeds weight limit (3000kg >

nt.

Strategized optimal cargo placement across aircraft zones.

This gives us these potential arrangements when B + A are in the Rear:

1. Front: E, Middle: C+D (C is H1, D is H3 - invalid)
2. Front: C, Middle: E+D (C is H1, D is H3 - invalid)
3. Front: D, Middle: E+C (A is H1 in Rear, D is H3 in Front, C is H1 in Middle - both A and C are not adjacent to D, so this is valid!)
4. Front: E+C, Middle: D (C is H1, D is H3 - invalid)

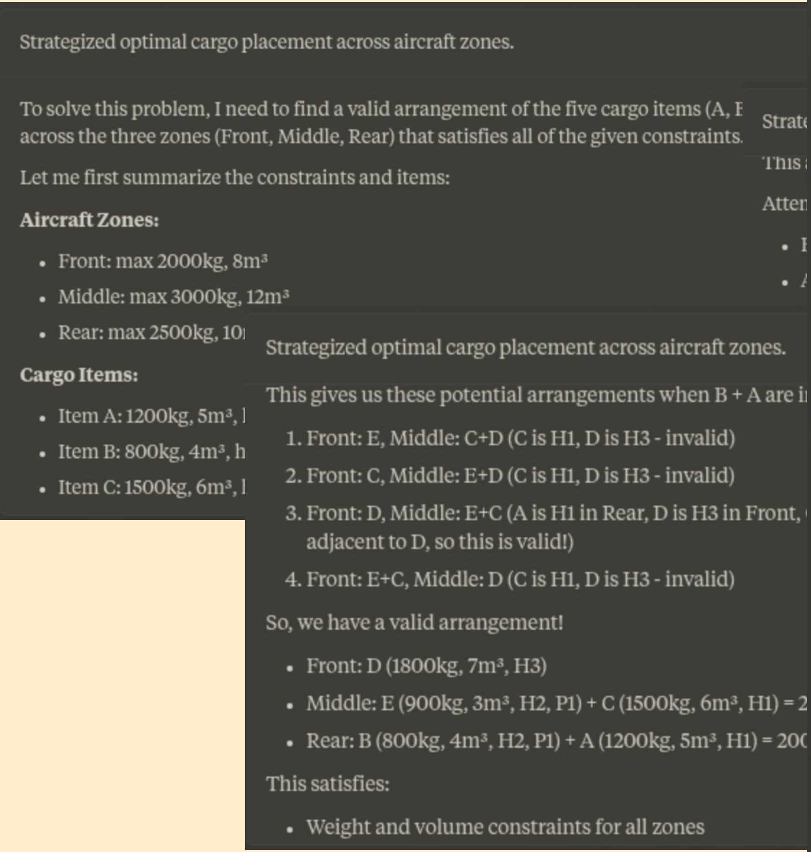
So, we have a valid arrangement!

- Front: D (1800kg, 7m^3, H3)
- Middle: E (900kg, 3m^3, H2, P1) + C (1500kg, 6m^3, H1) = 2400kg, 9m^3
- Rear: B (800kg, 4m^3, H2, P1) + A (1200kg, 5m^3, H1) = 2000kg, 9m^3

This satisfies:

- Weight and volume constraints for all zones

Verbalized Reasoning



Strategized optimal cargo placement across aircraft zones.

To solve this problem, I need to find a valid arrangement of the five cargo items (A,
across the three zones (Front, Middle, Rear) that satisfies all of the given constraints.

Let me first summarize the constraints and items:

Aircraft Zones:

- Front: max 2000kg, 8m^3
- Middle: max 3000kg, 12m^3
- Rear: max 2500kg, 10

Cargo Items:

- Item A: 1200kg, 5m^3,
- Item B: 800kg, 4m^3, h
- Item C: 1500kg, 6m^3,

Strate

This

Atten

Strategized optimal cargo placement across aircraft zones.

This gives us these potential arrangements when B + A are i

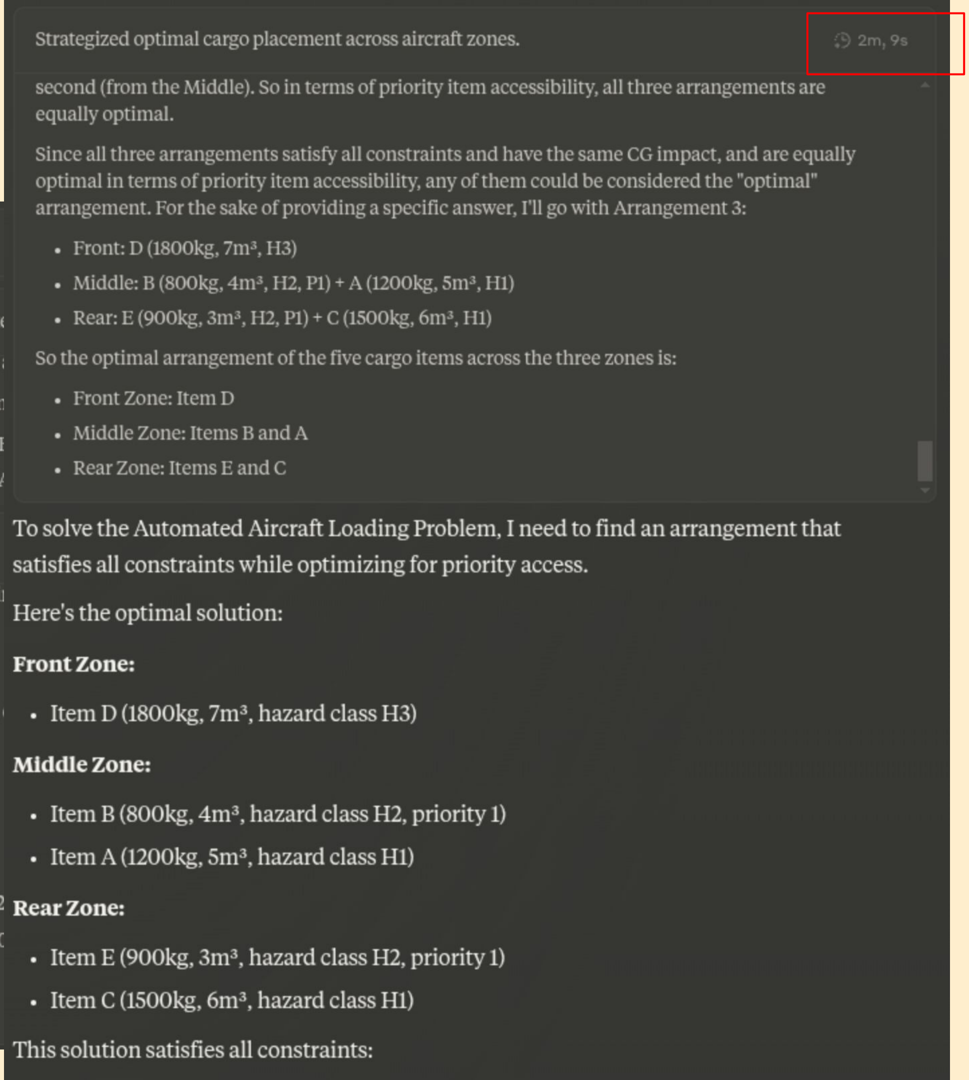
1. Front: E, Middle: C+D (C is H1, D is H3 - invalid)
2. Front: C, Middle: E+D (C is H1, D is H3 - invalid)
3. Front: D, Middle: E+C (A is H1 in Rear, D is H3 in Front,
adjacent to D, so this is valid!)
4. Front: E+C, Middle: D (C is H1, D is H3 - invalid)

So, we have a valid arrangement!

- Front: D (1800kg, 7m^3, H3)
- Middle: E (900kg, 3m^3, H2, P1) + C (1500kg, 6m^3, H1) = 2
- Rear: B (800kg, 4m^3, H2, P1) + A (1200kg, 5m^3, H1) = 200

This satisfies:

- Weight and volume constraints for all zones



Strategized optimal cargo placement across aircraft zones.

second (from the Middle). So in terms of priority item accessibility, all three arrangements are equally optimal.

Since all three arrangements satisfy all constraints and have the same CG impact, and are equally optimal in terms of priority item accessibility, any of them could be considered the "optimal" arrangement. For the sake of providing a specific answer, I'll go with Arrangement 3:

- Front: D (1800kg, 7m^3, H3)
- Middle: B (800kg, 4m^3, H2, P1) + A (1200kg, 5m^3, H1)
- Rear: E (900kg, 3m^3, H2, P1) + C (1500kg, 6m^3, H1)

So the optimal arrangement of the five cargo items across the three zones is:

- Front Zone: Item D
- Middle Zone: Items B and A
- Rear Zone: Items E and C

To solve the Automated Aircraft Loading Problem, I need to find an arrangement that satisfies all constraints while optimizing for priority access.

Here's the optimal solution:

Front Zone:

- Item D (1800kg, 7m^3, hazard class H3)

Middle Zone:

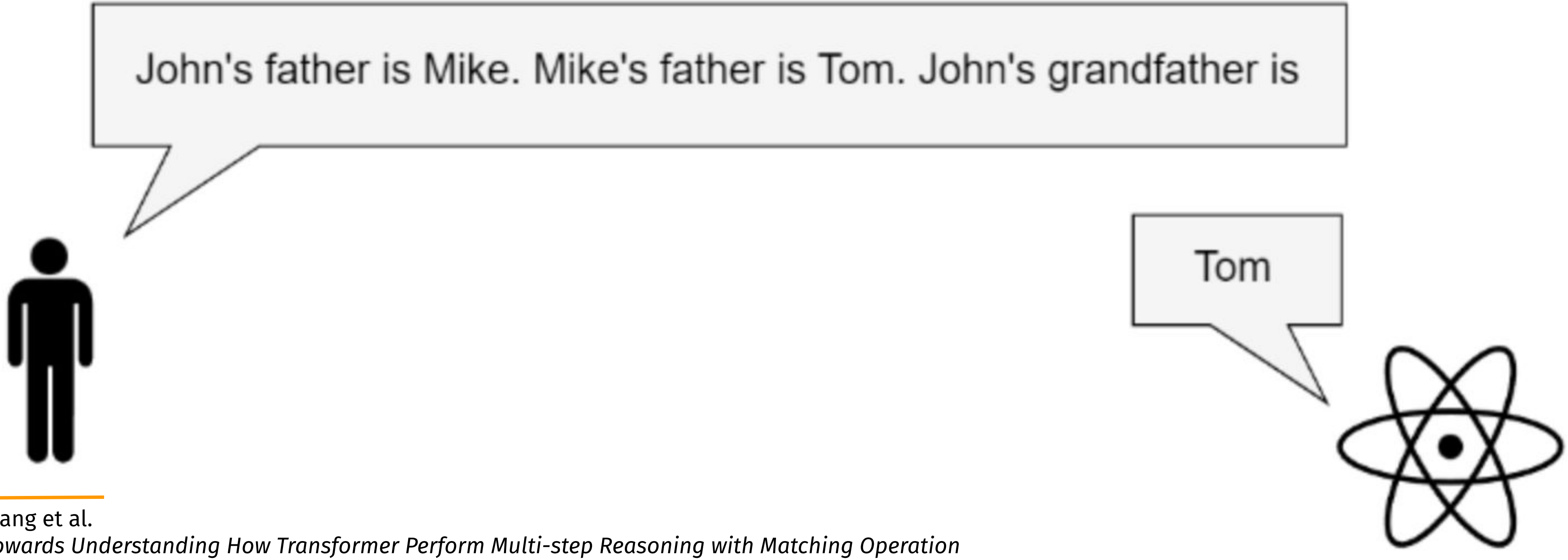
- Item B (800kg, 4m^3, hazard class H2, priority 1)
- Item A (1200kg, 5m^3, hazard class H1)

Rear Zone:

- Item E (900kg, 3m^3, hazard class H2, priority 1)
- Item C (1500kg, 6m^3, hazard class H1)

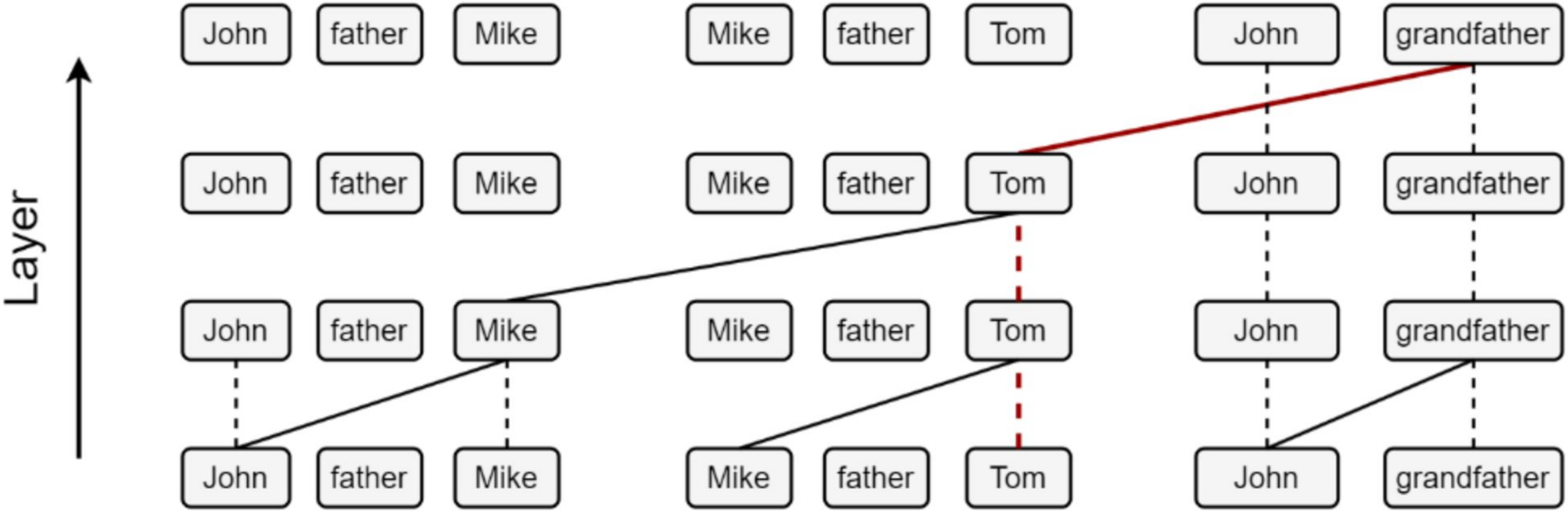
This solution satisfies all constraints:

Why is this n


John's father is Mike. Mike's father is Tom. John's grandfather is
Tom

Wang et al.
Towards Understanding How Transformer Perform Multi-step Reasoning with Matching Operation

What is the computational Depth of Transformers



Continuous Reasoning?

Moving the reasoning chain into the model's representation space.

A Recurrent-Depth Approach

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

Universal Transformers

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

Universal Transformers

Looped Transformers

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

Universal Transformers

Representation Recycling

Looped Transformers

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

Universal Transformers

RNNs with Adaptive Computation Time

Representation Recycling

Looped Transformers

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

Universal Transformers

RNNs with Adaptive Computation Time

Equilibrium Models

Representation Recycling

Looped Transformers

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

Universal Transformers

RNNs with Adaptive Computation Time

Equilibrium Models

Representation Recycling

Looped Transformers

Diffusion Models

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

Universal Transformers

RNNs with Adaptive Computation Time

Equilibrium Models

Representation Recycling

Looped Transformers

Diffusion Models

Iterative Refinement

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

Universal Transformers

RNNs with Adaptive Computation Time

Equilibrium Models

Representation Recycling

Looped Transformers

Diffusion Models

Iterative Refinement

Implicit Neural nets

What is recurrent depth?

A model that recurs "in depth" re-uses its layers or other subcomponents to create a deeper or shallower model.

Universal Transformers

RNNs with Adaptive Computation Time

Hopfield-Amari Models?

Equilibrium Models

Representation Recycling

Looped Transformers

Diffusion Models

Implicit Neural nets

Iterative Refinement

Why use recurrent-depth as a framework for test-time compute?

- No supervision on intermediate steps, so no CoT data needed with the right training objective
- No increased context length, linear complexity increase
- Recurrent-depth models have less parameters

- Recurrent-depth models are *compute-heavy*

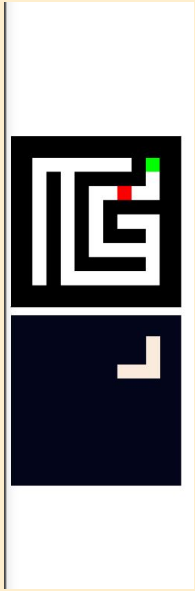
A good prior for "Reasoning"?

- Easy to learn iterative *algorithms*
- Harder for the model to memorize

Bansal*, Schwarzschild*, Borgnia, Emam, Huang, Goldblum, Goldstein
End-to-end Algorithm Synthesis with Recurrent Networks: Logical Extrapolation Without Overthinking

A good prior for "Reasoning"?

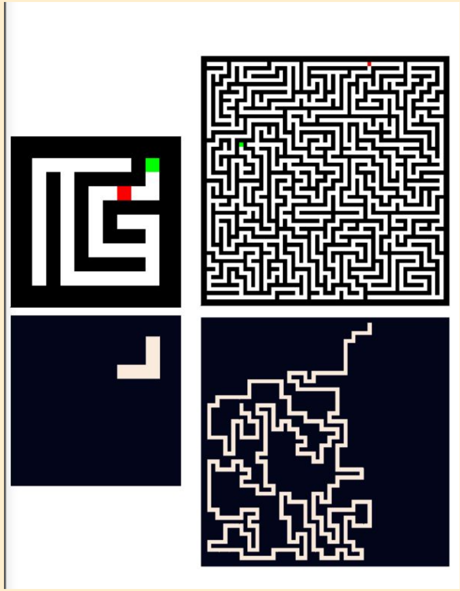
- Easy to learn iterative *algorithms*
- Harder for the model to memorize



Bansal*, Schwarzschild*, Borgnia, Emam, Huang, Goldblum, Goldstein
End-to-end Algorithm Synthesis with Recurrent Networks: Logical Extrapolation Without Overthinking

A good prior for "Reasoning"?

- Easy to learn iterative *algorithms*
- Harder for the model to memorize

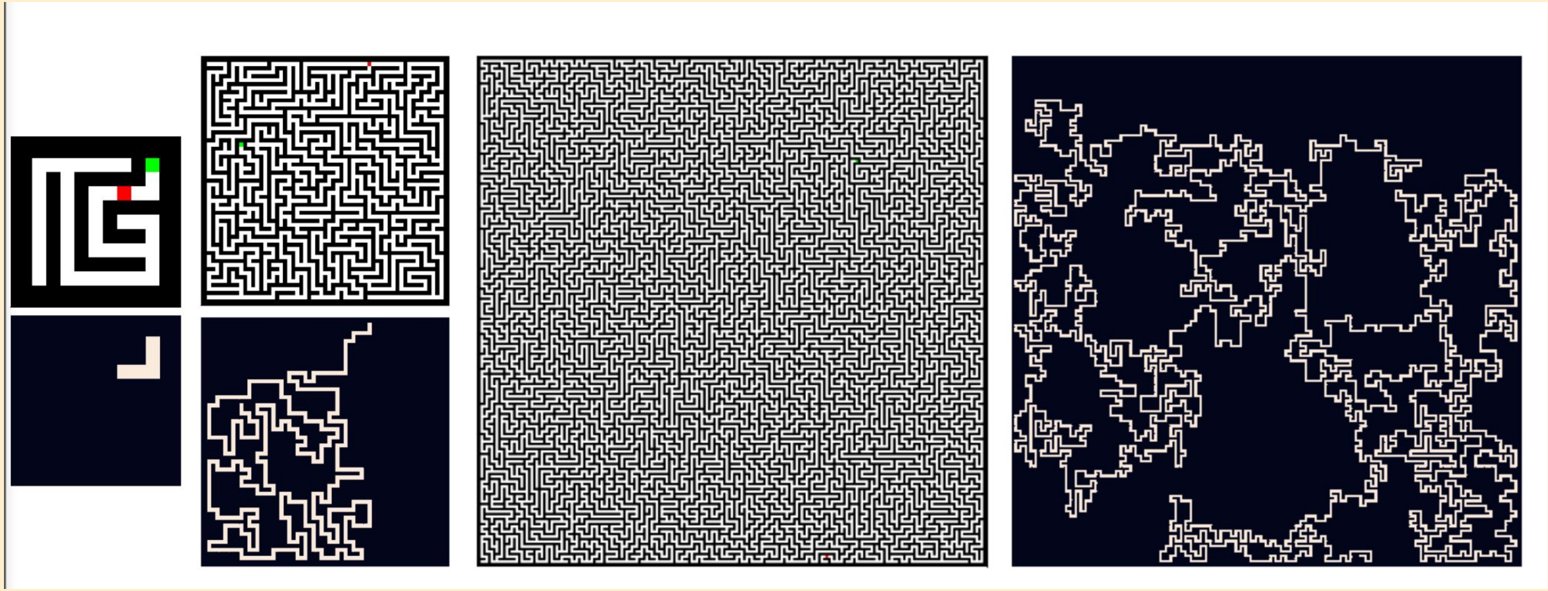


Bansal*, Schwarzschild*, Borgnia, Emam, Huang, Goldblum, Goldstein

End-to-end Algorithm Synthesis with Recurrent Networks: Logical Extrapolation Without Overthinking

A good prior for "Reasoning"?

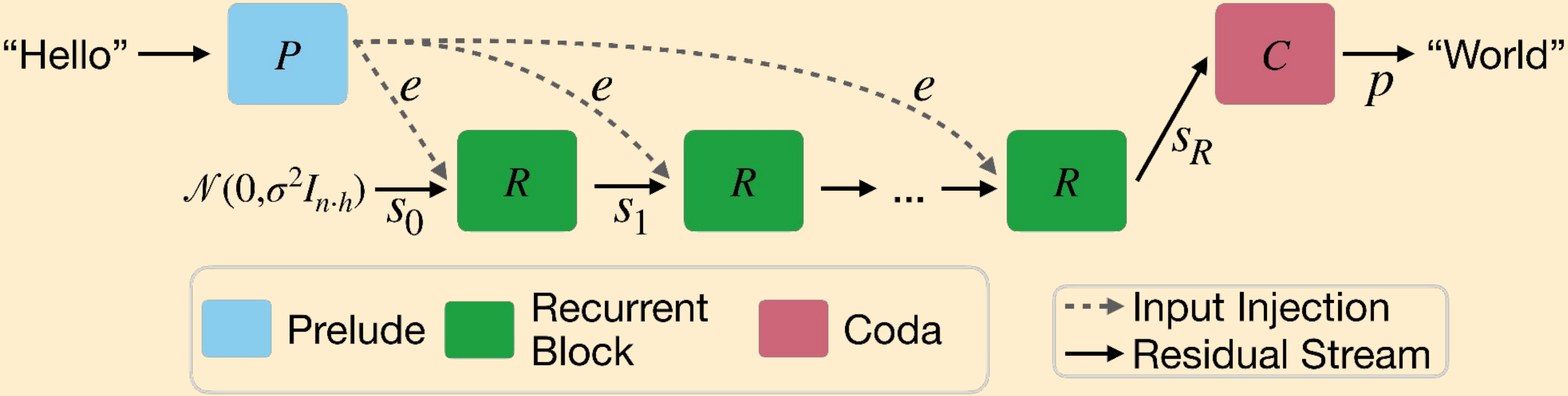
- Easy to learn iterative *algorithms*
- Harder for the model to memorize



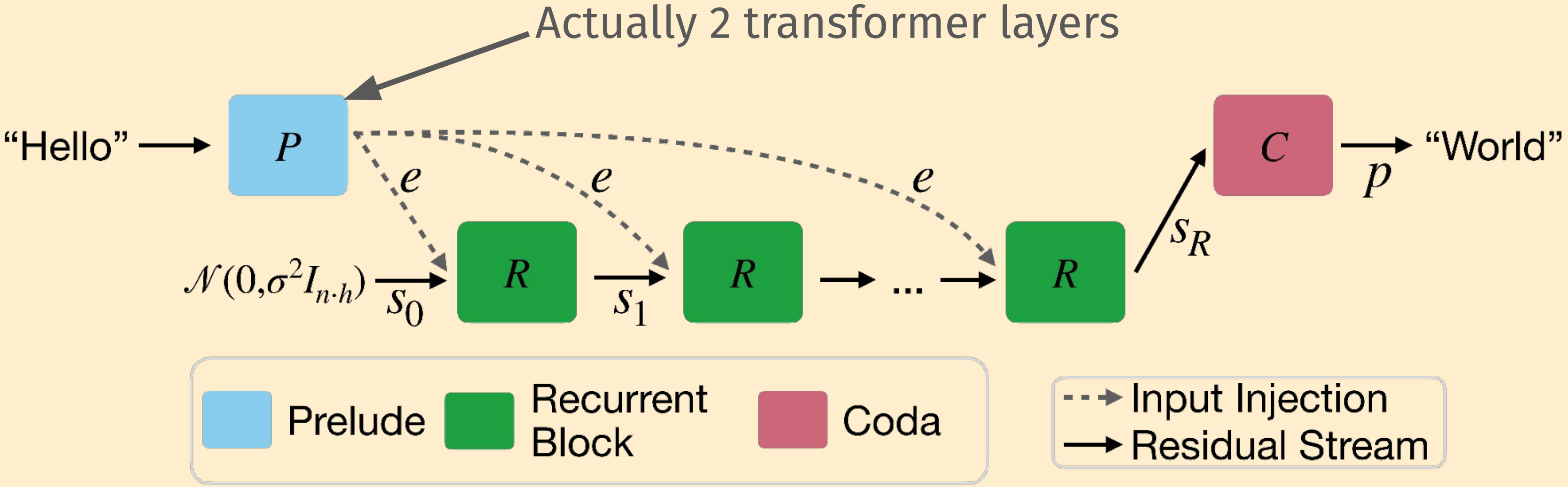
Bansal*, Schwarzschild*, Borgnia, Emam, Huang, Goldblum, Goldstein
End-to-end Algorithm Synthesis with Recurrent Networks: Logical Extrapolation Without Overthinking

A scalable recurrent (depth) architecture

Why did we call this *latent* recurrent depth?



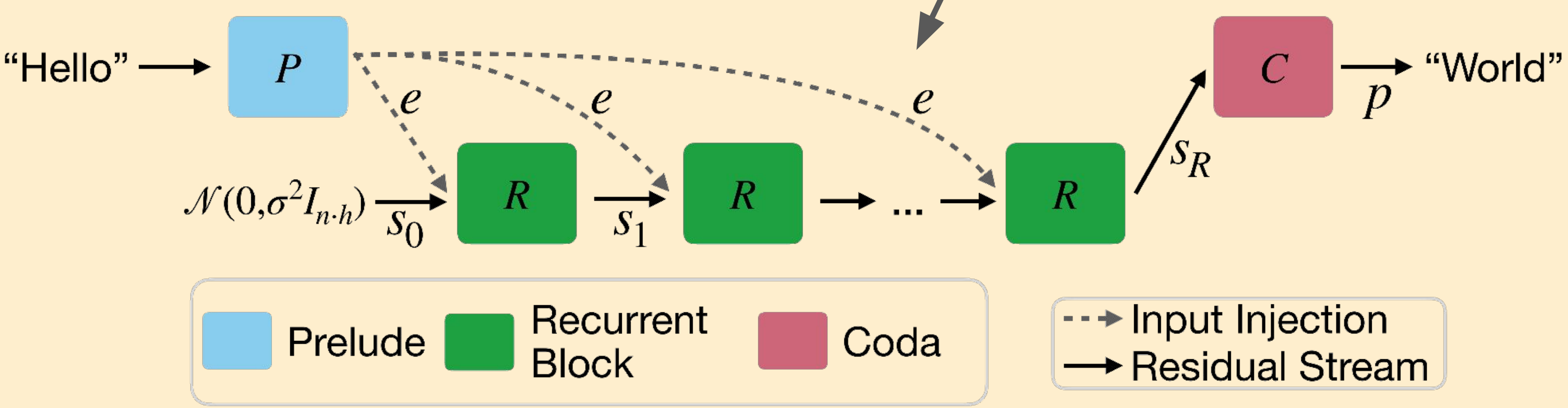
Why did we call this *latent* recurrent depth?



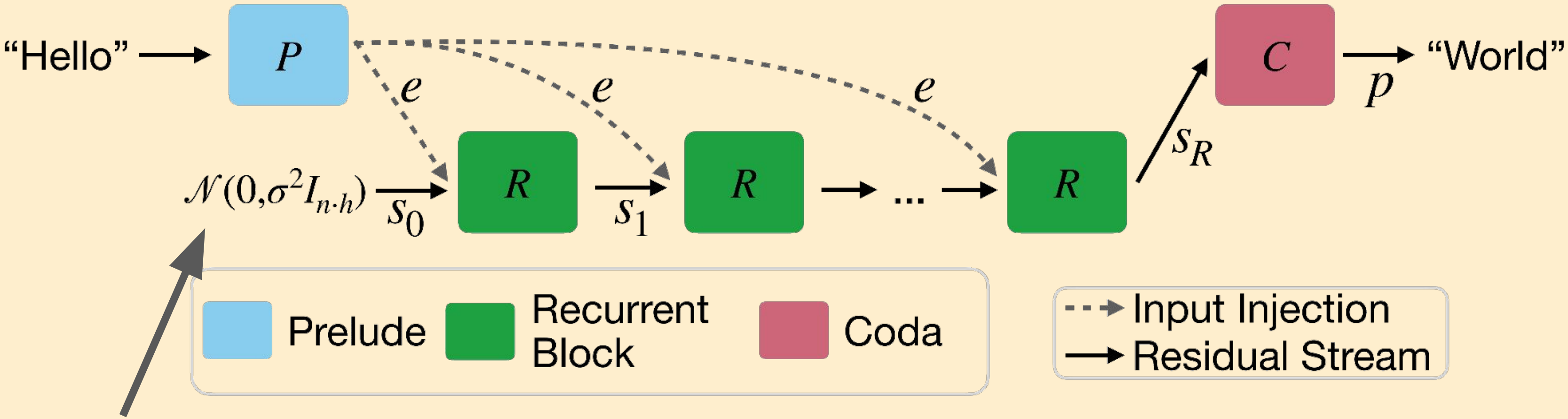
Actually 2 transformer layers

Why did we call this *latent* recurrent depth?

The "input injection" is crucial for stability



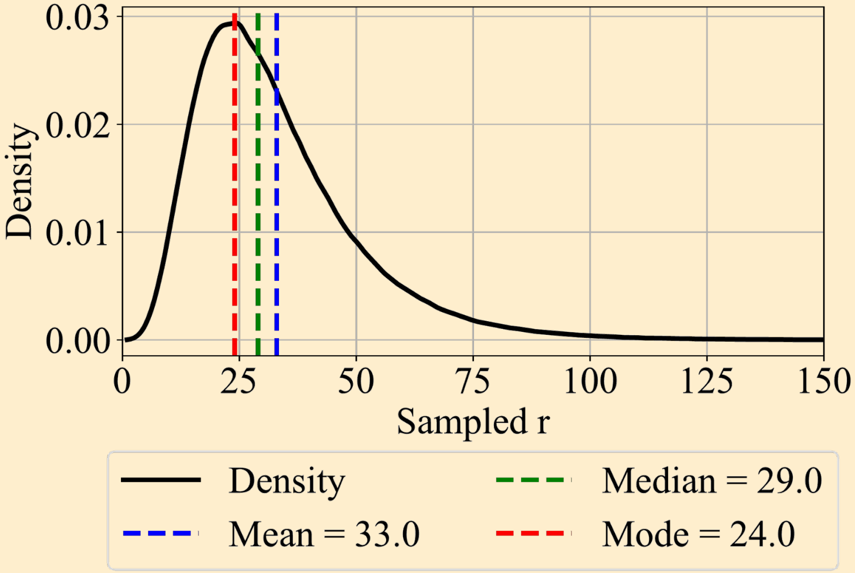
Why did we call this *latent* recurrent depth?



Diffusion Model connection, appears based on findings for path independence

What does that mean?

Training Objective



$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x}\in X}\mathbb{E}_{r\sim\Lambda} L\left(m_\theta(\mathbf{x}, r), \mathbf{x}'\right)$$

For every training sequence

- Sample a random* number of steps r to recur
- Compute r forward steps
- Compute loss based on the last k steps.

Training Objective Alternatives?

- **Universal Transformers:** Train with halting module
- **Equilibrium Models:** Iterate to convergence, differentiate fixed point based on IFT
- **Weight-shared models:** Just fix number of steps, train as normal
- **Diffusion Model:** Train to denoise target hidden states

Actually Training a Model at Scale

How do we show that this *actually* scales?

- 100m parameter, 10B tokens prototypes all work great…

How do we show that this *actually* scales?

- 100m parameter, 10B tokens prototypes all work great..

To show that this worked we define a more convincing

target, ***Huginn-3.5B***

How do we show that this *actually* scales?

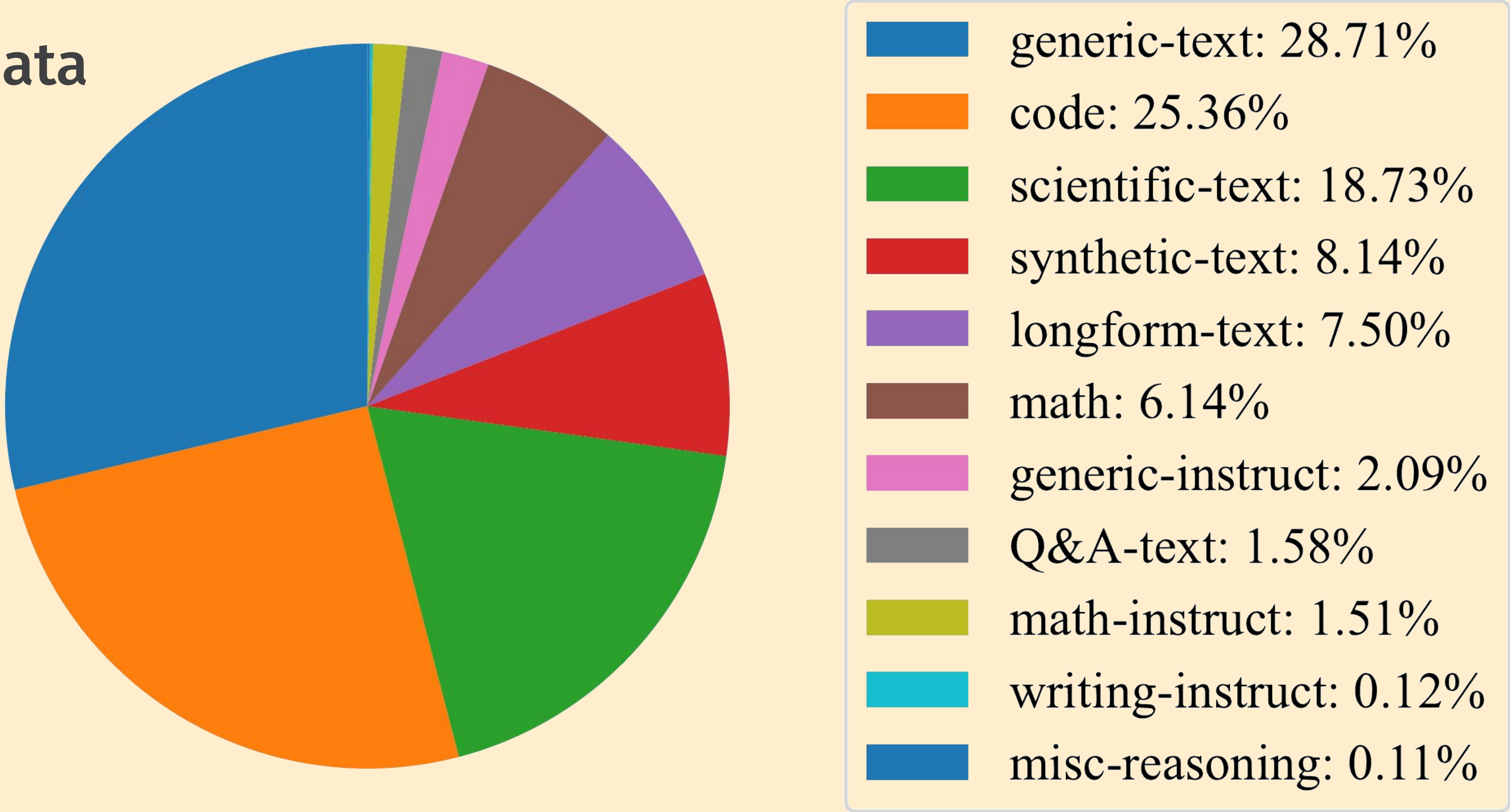
- 100m parameter, 10B tokens prototypes all work great...

To show that this worked we define a more convincing

target, ***Huginn-3.5B***

- 2 + 4 + 2 layers, tied embeds, 3.5B parameters
- We target 1T tokens of a wide pretraining mix
- Will it actually train to be a (semi)-competitive language and reasoning model?
- Who will give us compute ...

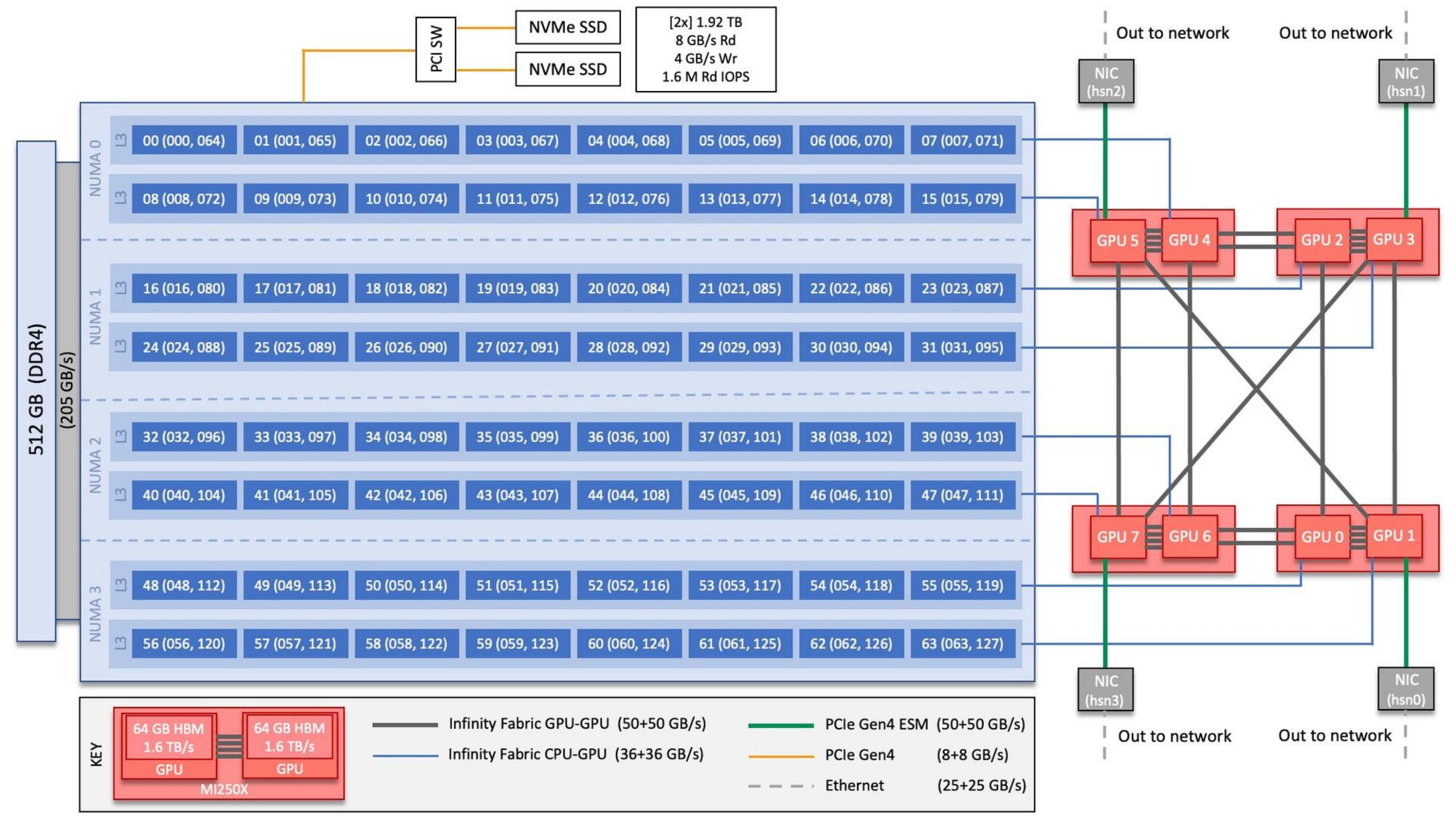
Data


generic-text: 28.71%
code: 25.36%
scientific-text: 18.73%
synthetic-text: 8.14%
longform-text: 7.50%
math: 6.14%
generic-instruct: 2.09%
Q&A-text: 1.58%
math-instruct: 1.51%
writing-instruct: 0.12%
misc-reasoning: 0.11%

Hardware - The *Frontier* Supercomputer

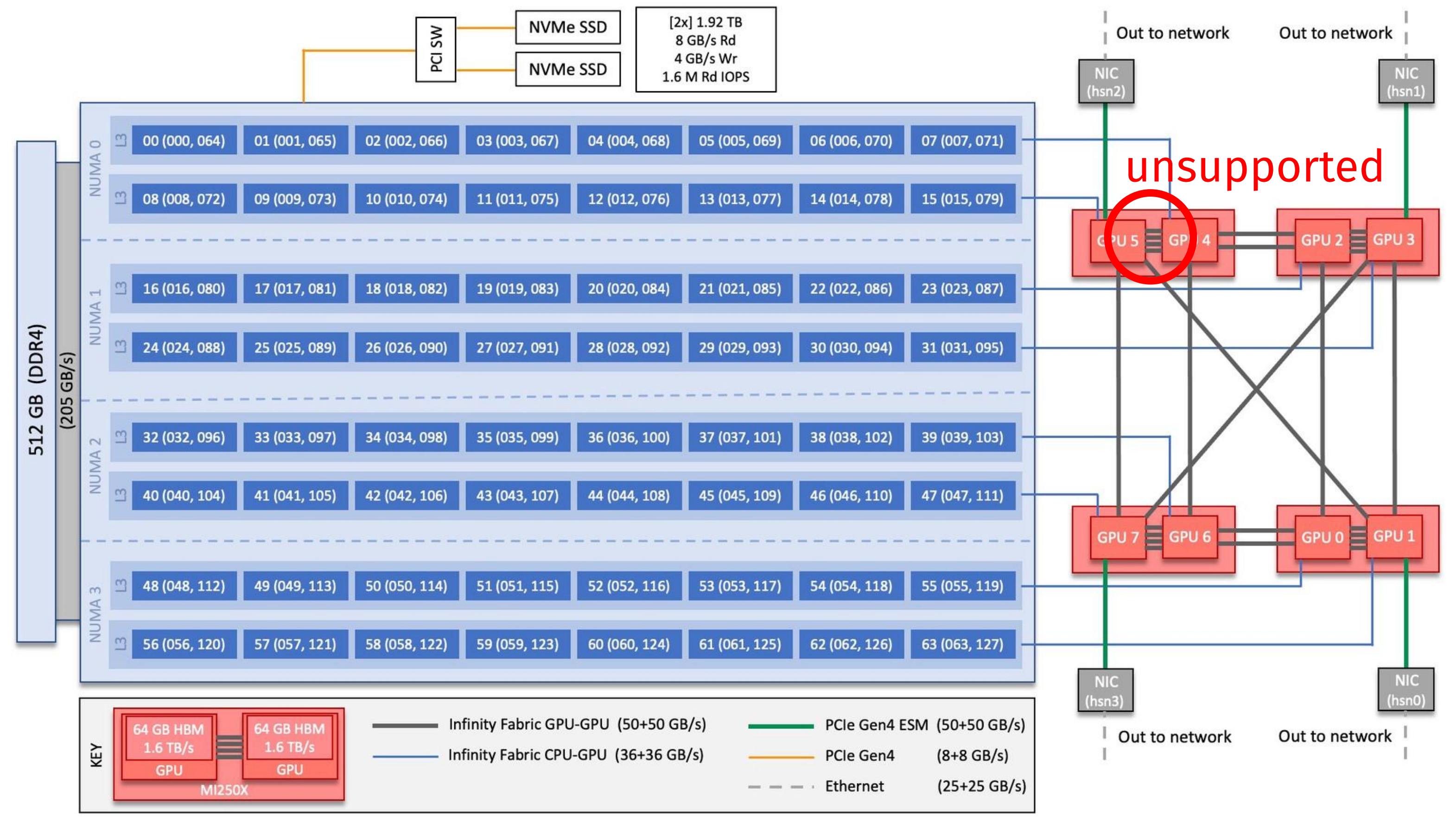
- Oak Ridge National Labs Exascale Machine
- Compute via the INCITE program
- 8x MI250X **AMD** previous-generation cards per node
- Nominally high inter-node connects

Hardware - The *Frontier* Supercomputer



NUMA 0	L3	00 (000, 064)	01 (001, 065)	02 (002, 066)	03 (003, 067)	04 (004, 068)	05 (005, 069)	06 (006, 070)	07 (007, 071)
	L3	08 (008, 072)	09 (009, 073)	10 (010, 074)	11 (011, 075)	12 (012, 076)	13 (013, 077)	14 (014, 078)	15 (015, 079)
NUMA 1	L3	16 (016, 080)	17 (017, 081)	18 (018, 082)	19 (019, 083)	20 (020, 084)	21 (021, 085)	22 (022, 086)	23 (023, 087)
	L3	24 (024, 088)	25 (025, 089)	26 (026, 090)	27 (027, 091)	28 (028, 092)	29 (029, 093)	30 (030, 094)	31 (031, 095)
NUMA 2	L3	32 (032, 096)	33 (033, 097)	34 (034, 098)	35 (035, 099)	36 (036, 100)	37 (037, 101)	38 (038, 102)	39 (039, 103)
	L3	40 (040, 104)	41 (041, 105)	42 (042, 106)	43 (043, 107)	44 (044, 108)	45 (045, 109)	46 (046, 110)	47 (047, 111)
NUMA 3	L3	48 (048, 112)	49 (049, 113)	50 (050, 114)	51 (051, 115)	52 (052, 116)	53 (053, 117)	54 (054, 118)	55 (055, 119)
	L3	56 (056, 120)	57 (057, 121)	58 (058, 122)	59 (059, 123)	60 (060, 124)	61 (061, 125)	62 (062, 126)	63 (063, 127)

Hardware - The *Frontier* Supercomputer

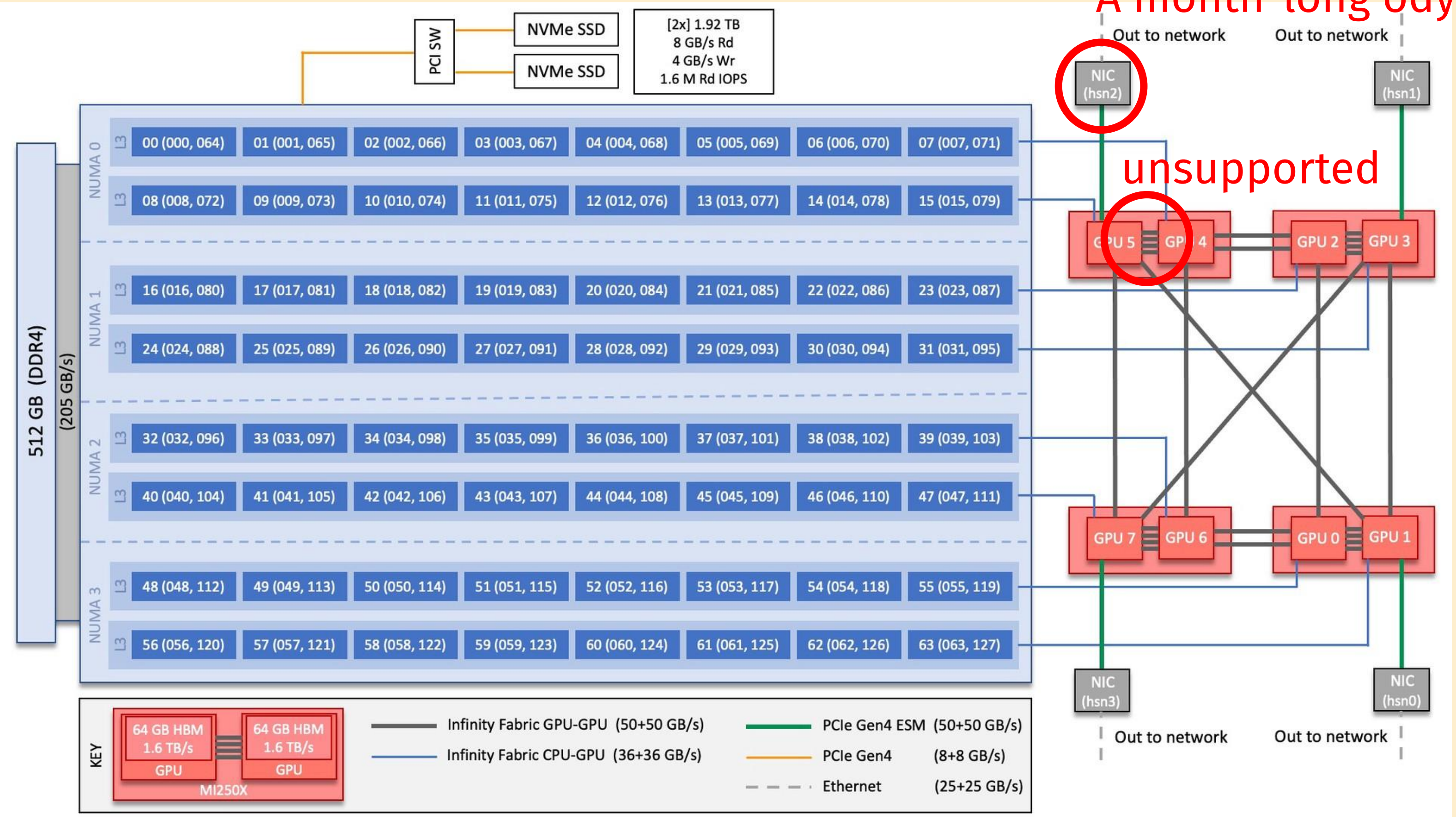


NUMA 0	L3	00 (000, 064)	01 (001, 065)	02 (002, 066)	03 (003, 067)	04 (004, 068)	05 (005, 069)	06 (006, 070)	07 (007, 071)
	L3	08 (008, 072)	09 (009, 073)	10 (010, 074)	11 (011, 075)	12 (012, 076)	13 (013, 077)	14 (014, 078)	15 (015, 079)
NUMA 1	L3	16 (016, 080)	17 (017, 081)	18 (018, 082)	19 (019, 083)	20 (020, 084)	21 (021, 085)	22 (022, 086)	23 (023, 087)
	L3	24 (024, 088)	25 (025, 089)	26 (026, 090)	27 (027, 091)	28 (028, 092)	29 (029, 093)	30 (030, 094)	31 (031, 095)
NUMA 2	L3	32 (032, 096)	33 (033, 097)	34 (034, 098)	35 (035, 099)	36 (036, 100)	37 (037, 101)	38 (038, 102)	39 (039, 103)
	L3	40 (040, 104)	41 (041, 105)	42 (042, 106)	43 (043, 107)	44 (044, 108)	45 (045, 109)	46 (046, 110)	47 (047, 111)
NUMA 3	L3	48 (048, 112)	49 (049, 113)	50 (050, 114)	51 (051, 115)	52 (052, 116)	53 (053, 117)	54 (054, 118)	55 (055, 119)
	L3	56 (056, 120)	57 (057, 121)	58 (058, 122)	59 (059, 123)	60 (060, 124)	61 (061, 125)	62 (062, 126)	63 (063, 127)

Infinity Fabric GPU-GPU	(50+50 GB/s)	PCIe Gen4 ESM	(50+50 GB/s)
Infinity Fabric CPU-GPU	(36+36 GB/s)	PCIe Gen4	(8+8 GB/s)
		Ethernet	(25+25 GB/s)

unsupported

Hardware - The *Frontier* Supercomputer



A month-long odyssey

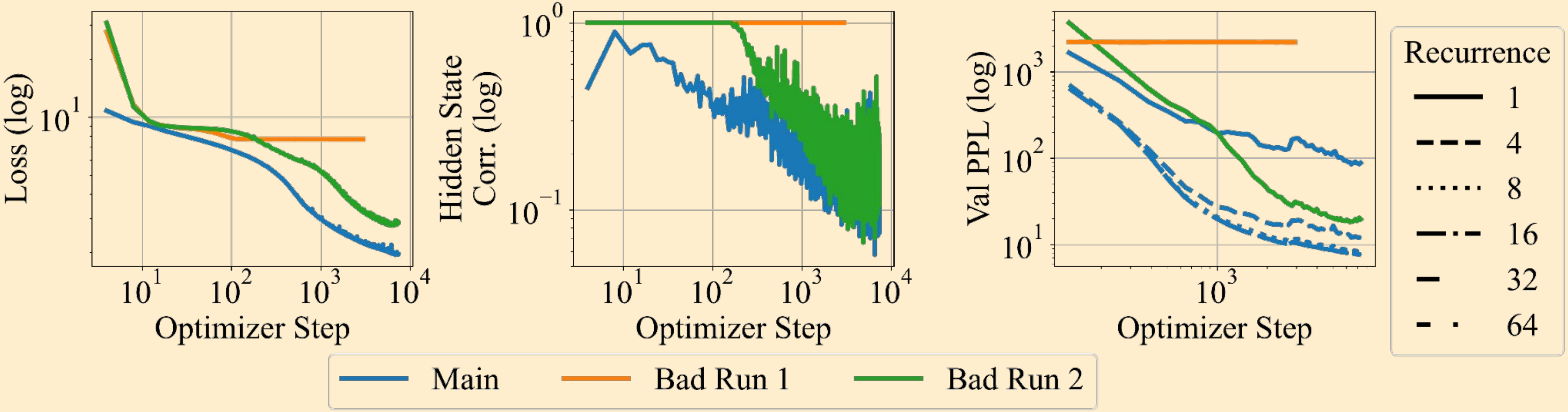
unsupported

Hardware - The *Frontier* Supercomputer

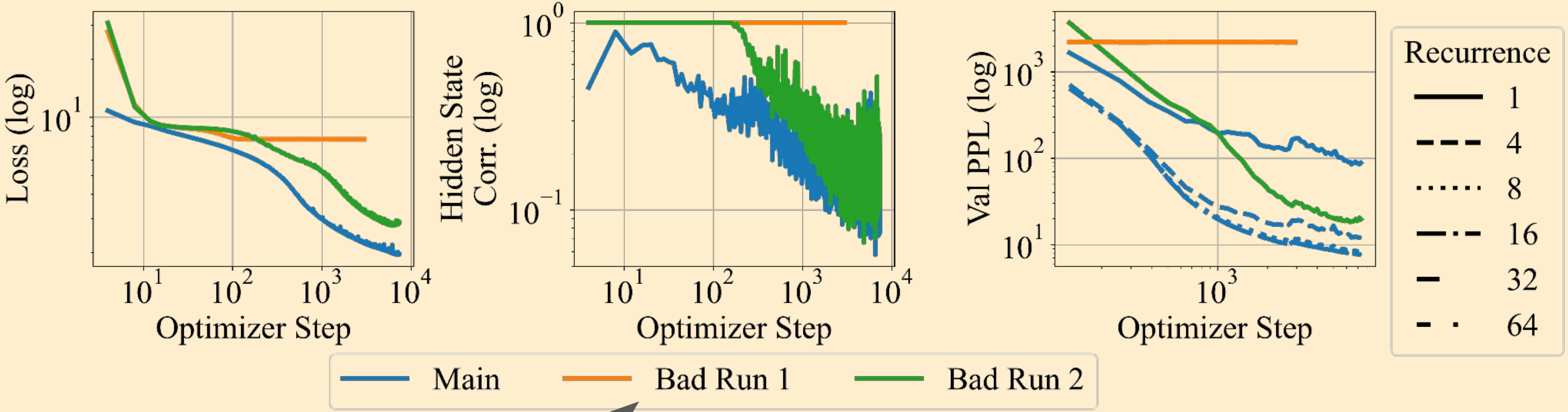
...
[Fun]
...

... We train on 4096 AMD GPUs in 21 segments of up to 12 hours with a constant learning rate with warmup. The setup is distributed data parallel with a batch size of 16m tokens.

Initial Scaling Issues

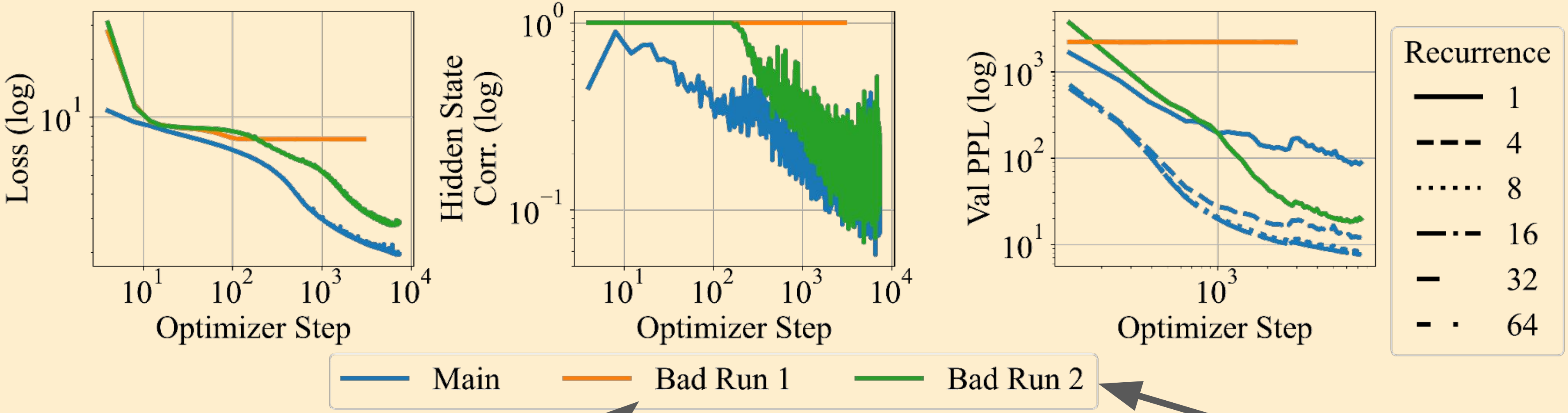


Initial Scaling Issues



Large LR, no embed scale,
no learned adapter

Initial Scaling Issues



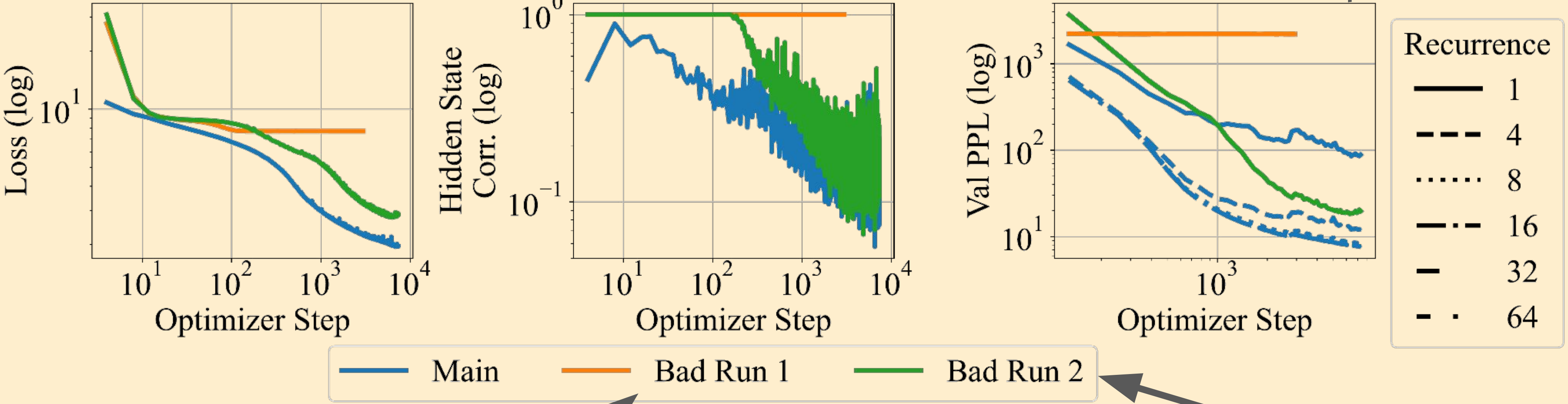
Large LR, no embed scale, no learned adapter

Large LR, prenorm layers, learned adapter

Initial Scaling Issues

Main Run:

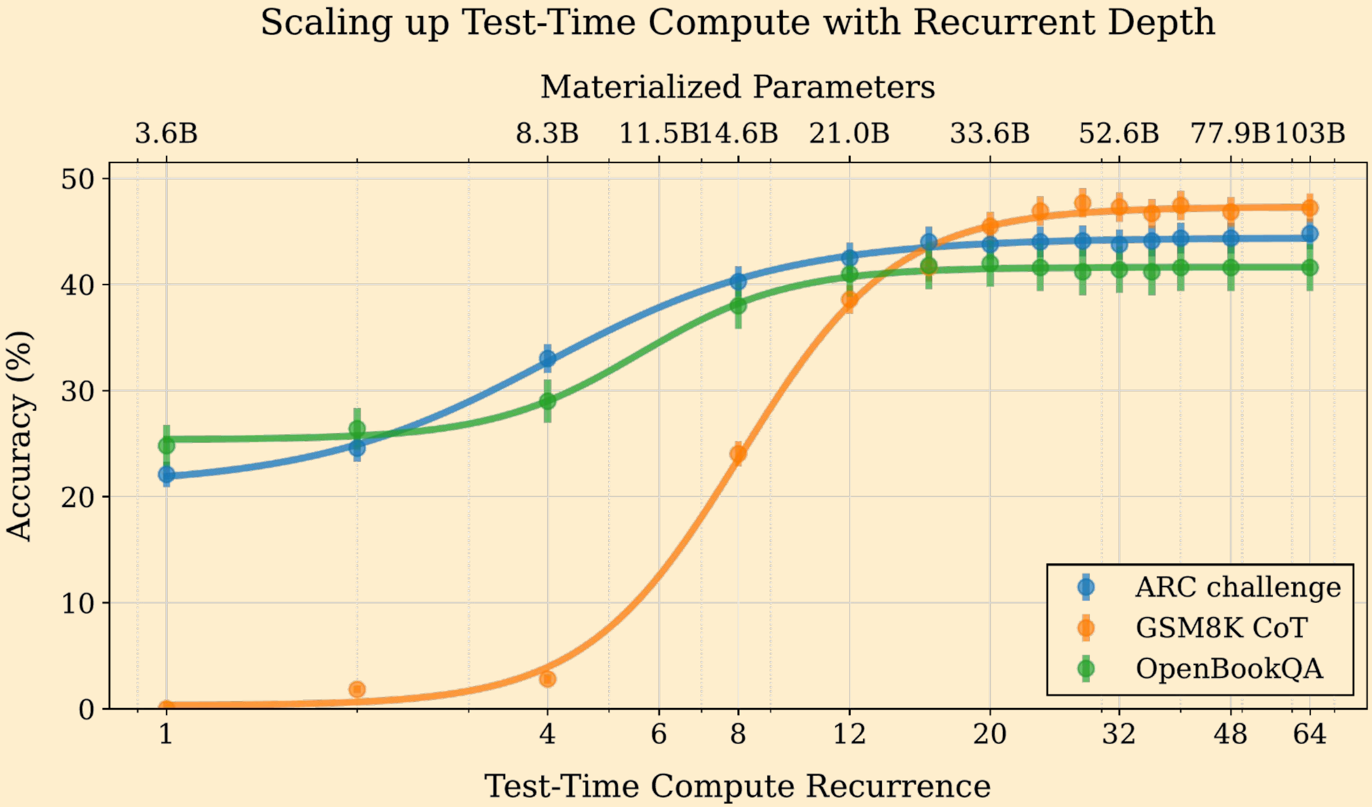
- Embed scale
- Smaller lr
- "Sandwich" double RMS normalization
- Learned Adapter



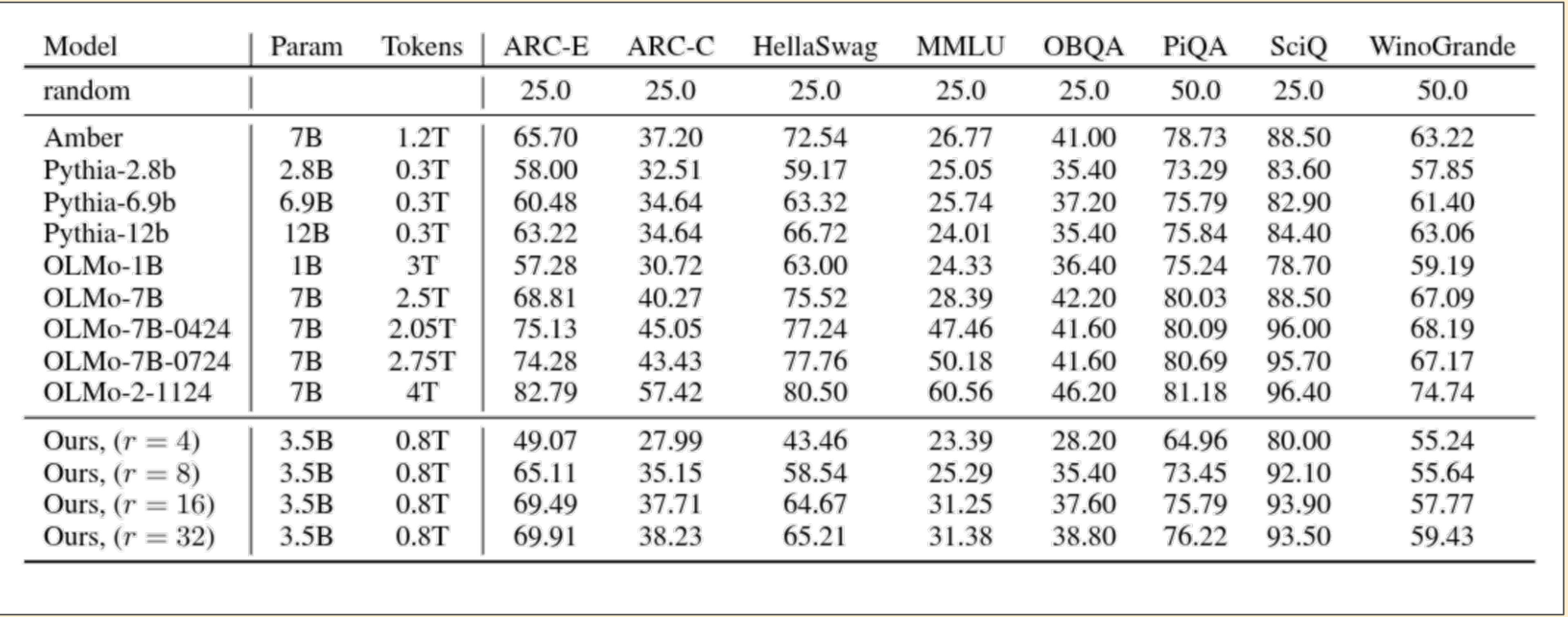
Large LR, no embed scale, no learned adapter

Large LR, prenorm layers, learned adapter

Results

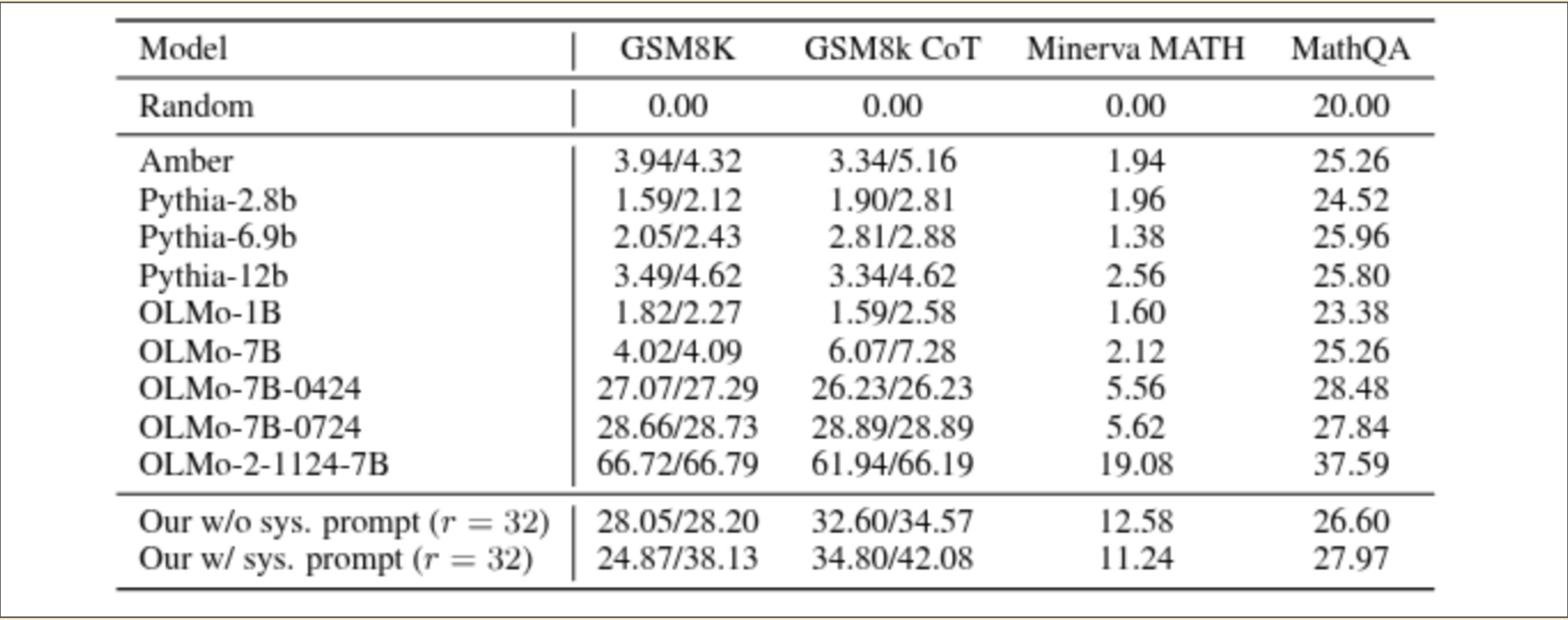

Scaling up Test-Time Compute with Recurrent Depth
Materialized Parameters
3.6B
8.3B
11.5B
14.6B
21.0B
33.6B
52.6B
77.9B
103B
Accuracy (%)
0
10
20
30
40
50
1
4
6
8
12
20
32
48
64
Test-Time Compute Recurrence
ARC challenge
GSM8K CoT
OpenBookQA

Standard benchmarks



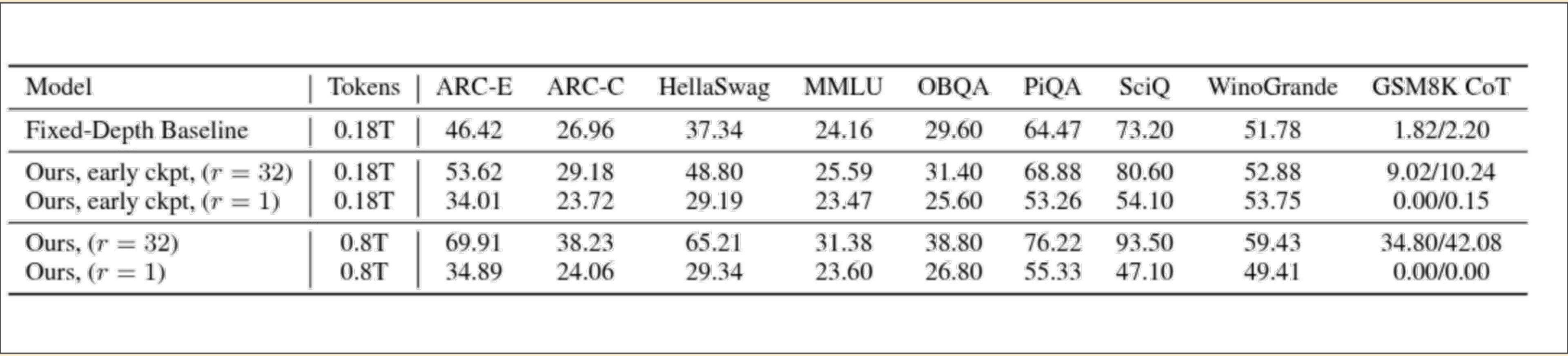
Model	Param	Tokens	ARC-E	ARC-C	HellaSwag	MMLU	OBQA	PiQA	SciQ	WinoGrande
random			25.0	25.0	25.0	25.0	25.0	50.0	25.0	50.0
Amber	7B	1.2T	65.70	37.20	72.54	26.77	41.00	78.73	88.50	63.22
Pythia-2.8b	2.8B	0.3T	58.00	32.51	59.17	25.05	35.40	73.29	83.60	57.85
Pythia-6.9b	6.9B	0.3T	60.48	34.64	63.32	25.74	37.20	75.79	82.90	61.40
Pythia-12b	12B	0.3T	63.22	34.64	66.72	24.01	35.40	75.84	84.40	63.06
OLMo-1B	1B	3T	57.28	30.72	63.00	24.33	36.40	75.24	78.70	59.19
OLMo-7B	7B	2.5T	68.81	40.27	75.52	28.39	42.20	80.03	88.50	67.09
OLMo-7B-0424	7B	2.05T	75.13	45.05	77.24	47.46	41.60	80.09	96.00	68.19
OLMo-7B-0724	7B	2.75T	74.28	43.43	77.76	50.18	41.60	80.69	95.70	67.17
OLMo-2-1124	7B	4T	82.79	57.42	80.50	60.56	46.20	81.18	96.40	74.74
Ours, ($r = 4$)	3.5B	0.8T	49.07	27.99	43.46	23.39	28.20	64.96	80.00	55.24
Ours, ($r = 8$)	3.5B	0.8T	65.11	35.15	58.54	25.29	35.40	73.45	92.10	55.64
Ours, ($r = 16$)	3.5B	0.8T	69.49	37.71	64.67	31.25	37.60	75.79	93.90	57.77
Ours, ($r = 32$)	3.5B	0.8T	69.91	38.23	65.21	31.38	38.80	76.22	93.50	59.43

Reasoning (grade-school math)



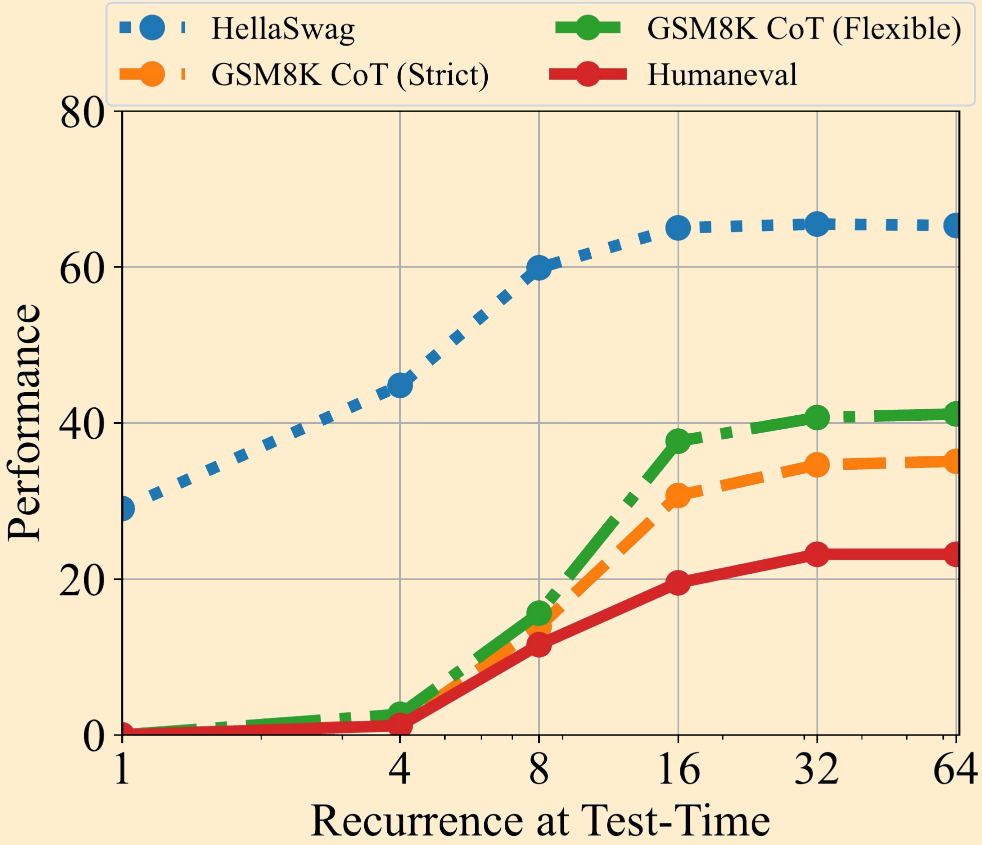
Model	GSM8K	GSM8k CoT	Minerva MATH	MathQA
Random	0.00	0.00	0.00	20.00
Amber	3.94/4.32	3.34/5.16	1.94	25.26
Pythia-2.8b	1.59/2.12	1.90/2.81	1.96	24.52
Pythia-6.9b	2.05/2.43	2.81/2.88	1.38	25.96
Pythia-12b	3.49/4.62	3.34/4.62	2.56	25.80
OLMo-1B	1.82/2.27	1.59/2.58	1.60	23.38
OLMo-7B	4.02/4.09	6.07/7.28	2.12	25.26
OLMo-7B-0424	27.07/27.29	26.23/26.23	5.56	28.48
OLMo-7B-0724	28.66/28.73	28.89/28.89	5.62	27.84
OLMo-2-1124-7B	66.72/66.79	61.94/66.19	19.08	37.59
Our w/o sys. prompt ($r = 32$)	28.05/28.20	32.60/34.57	12.58	26.60
Our w/ sys. prompt ($r = 32$)	24.87/38.13	34.80/42.08	11.24	27.97

Reasoning (grade-school math)

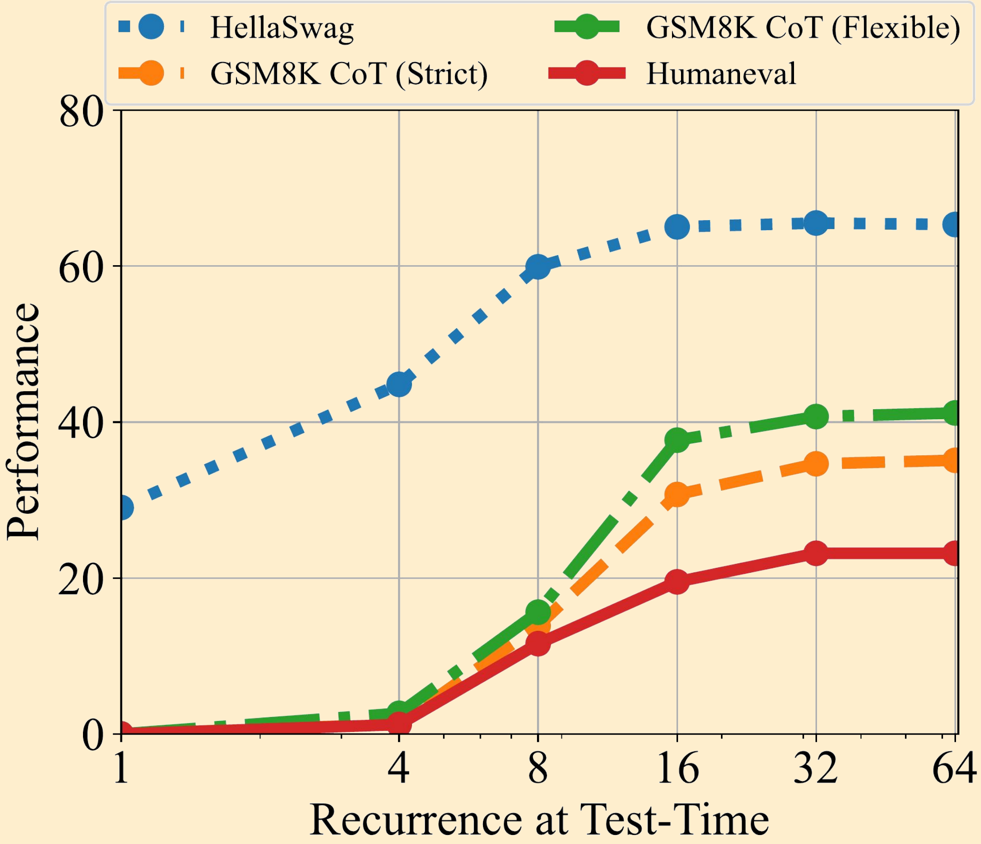


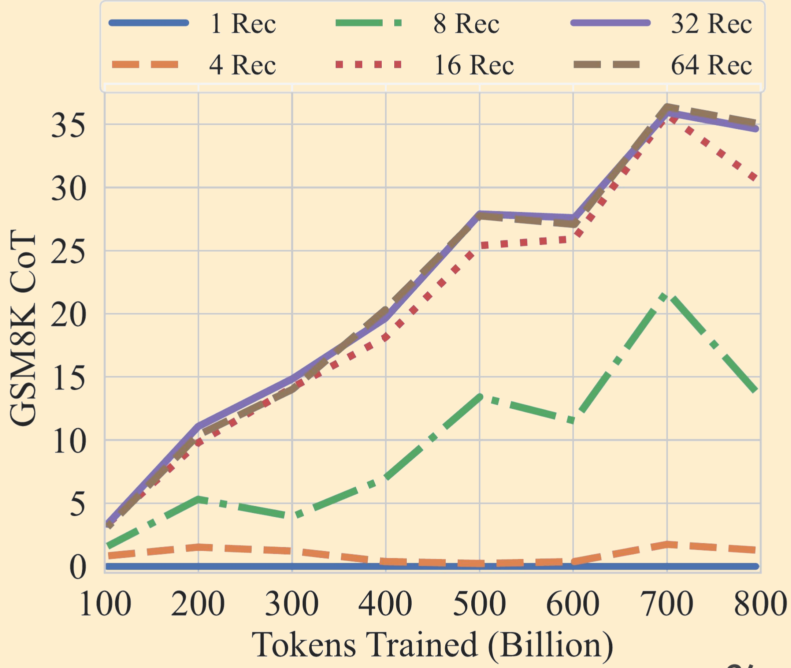
Model	Tokens	ARC-E	ARC-C	HellaSwag	MMLU	OBQA	PiQA	SciQ	WinoGrande	GSM8K CoT
Fixed-Depth Baseline	0.18T	46.42	26.96	37.34	24.16	29.60	64.47	73.20	51.78	1.82/2.20
Ours, early ckpt, ($r = 32$)	0.18T	53.62	29.18	48.80	25.59	31.40	68.88	80.60	52.88	9.02/10.24
Ours, early ckpt, ($r = 1$)	0.18T	34.01	23.72	29.19	23.47	25.60	53.26	54.10	53.75	0.00/0.15
Ours, ($r = 32$)	0.8T	69.91	38.23	65.21	31.38	38.80	76.22	93.50	59.43	34.80/42.08
Ours, ($r = 1$)	0.8T	34.89	24.06	29.34	23.60	26.80	55.33	47.10	49.41	0.00/0.00

Scaling Compute helps on harder tasks

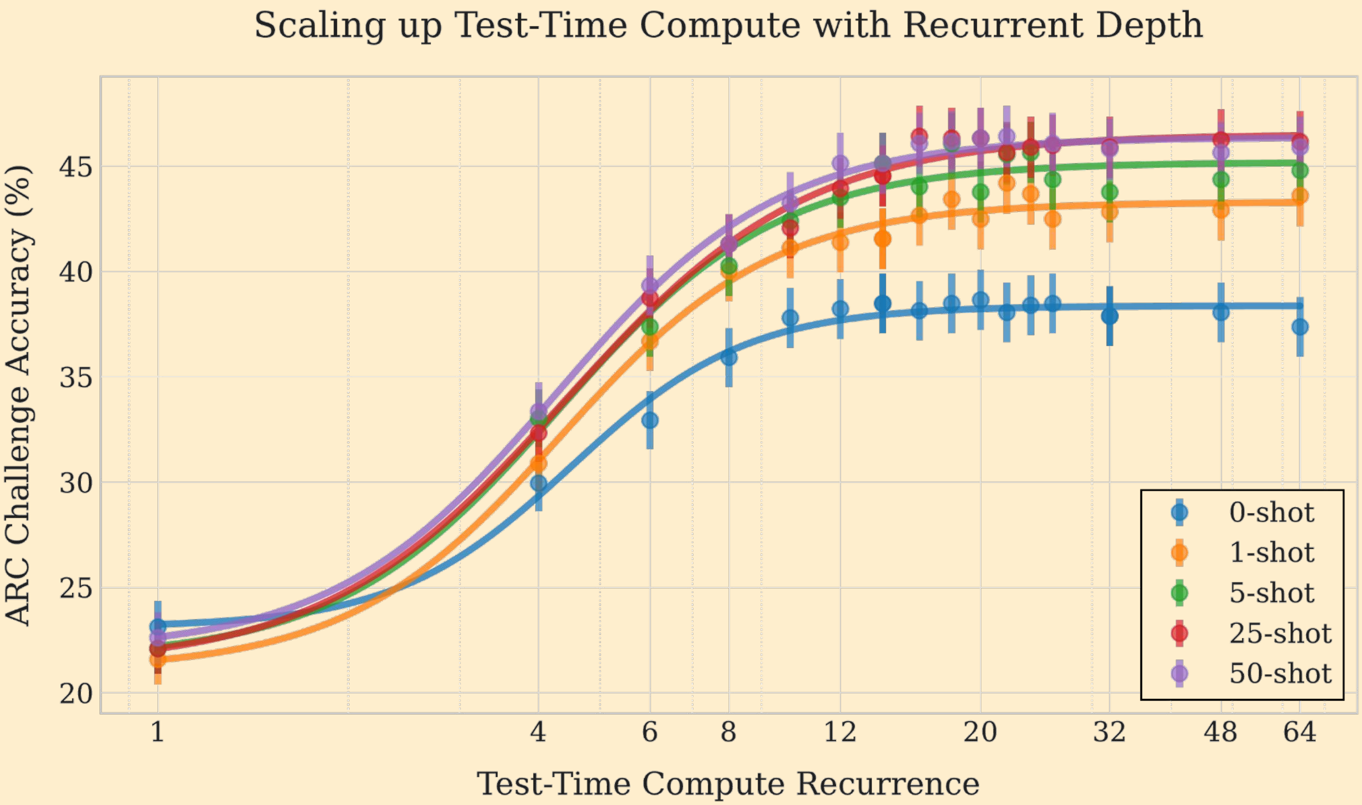


Scaling Test-Time Compute vs Scaling Pretraining



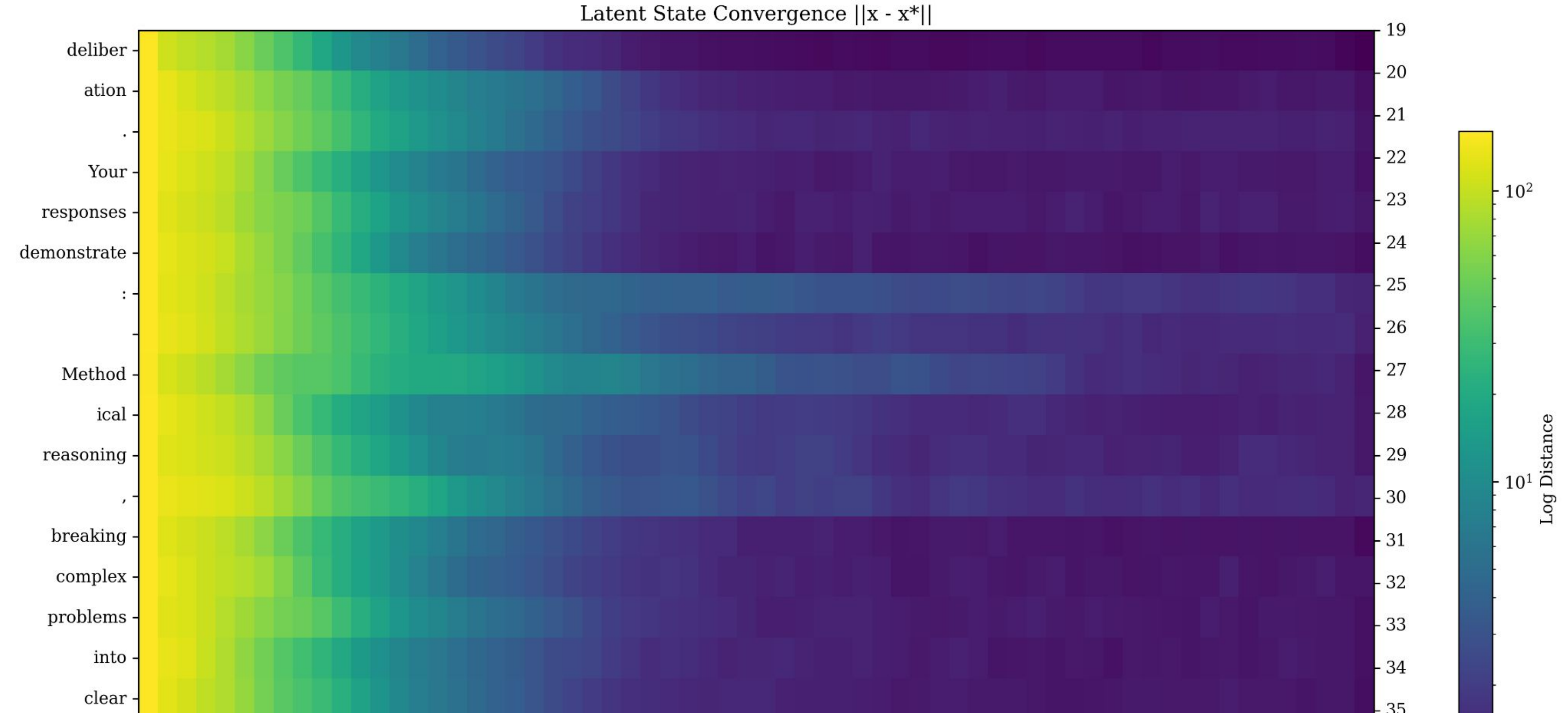


Scaling Context vs Scaling Test-Time Compute

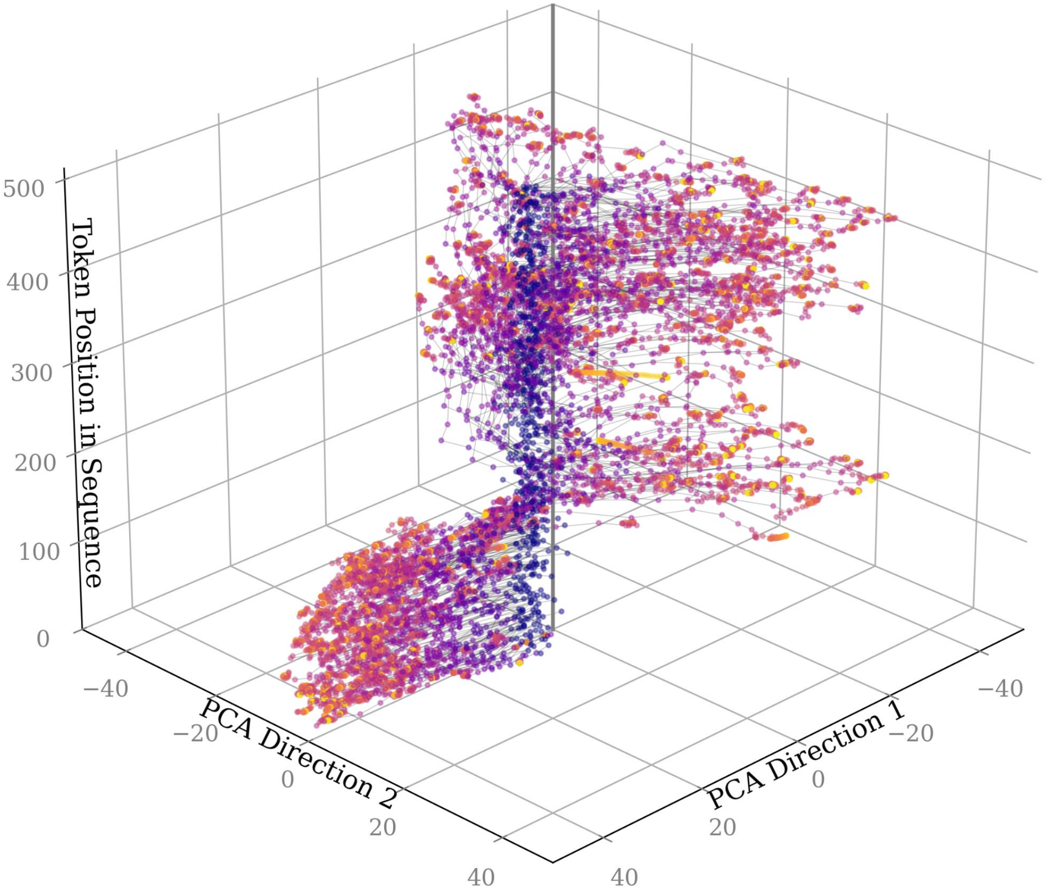


What is the model doing?

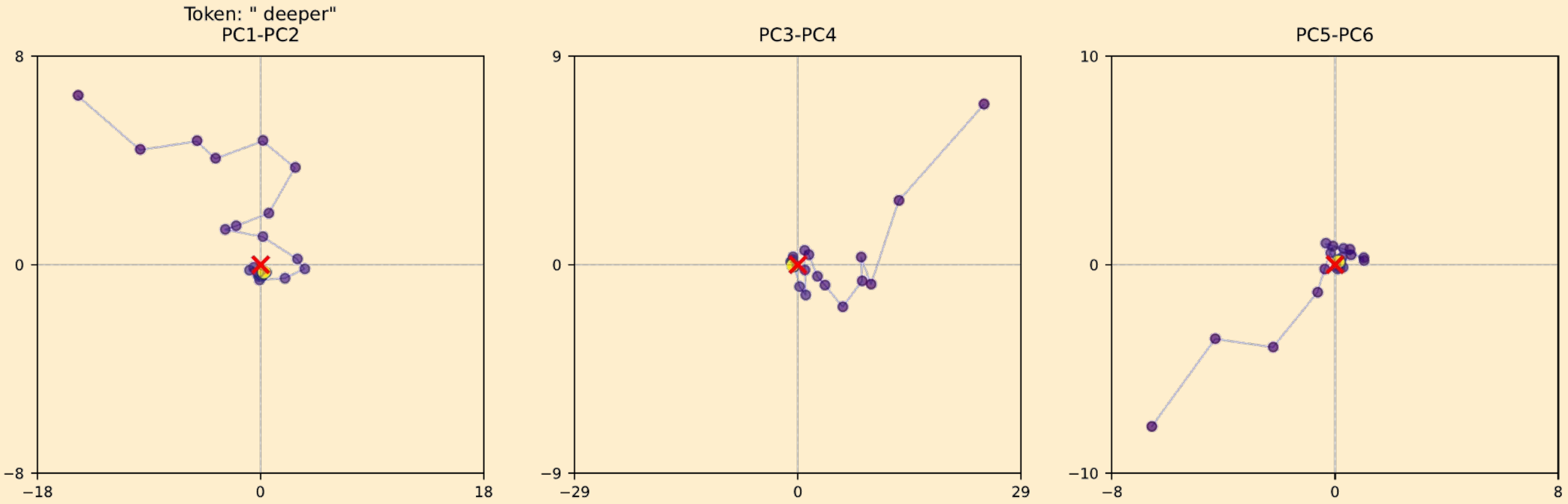
Convergence rates per recurrence step at every position



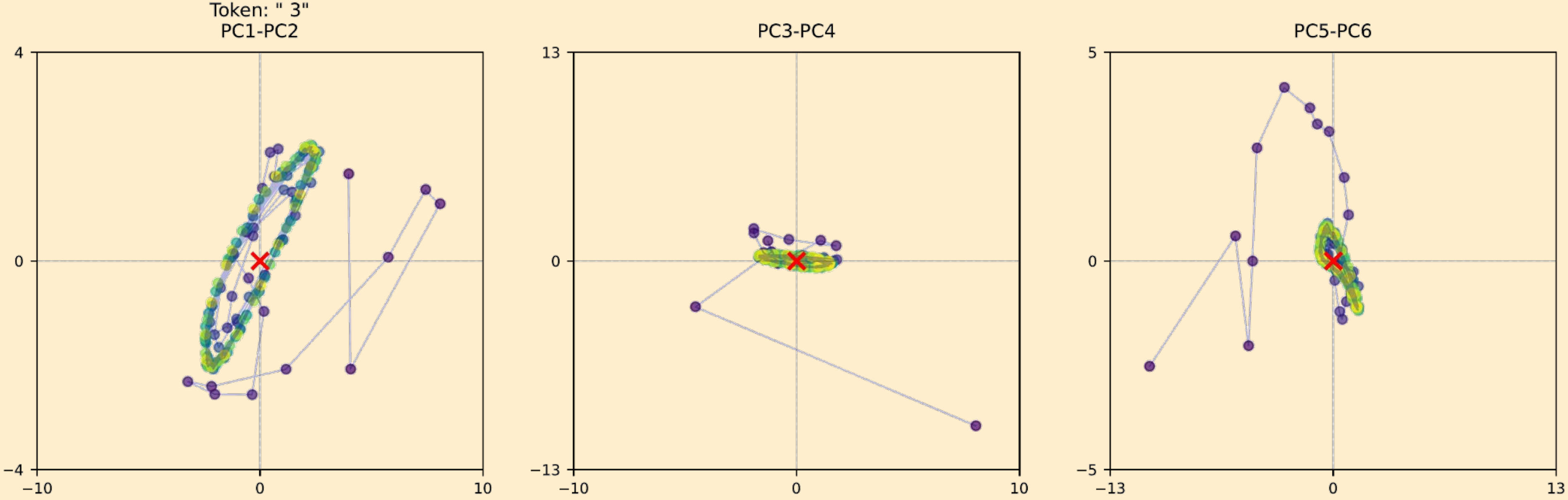
Token Trajectories



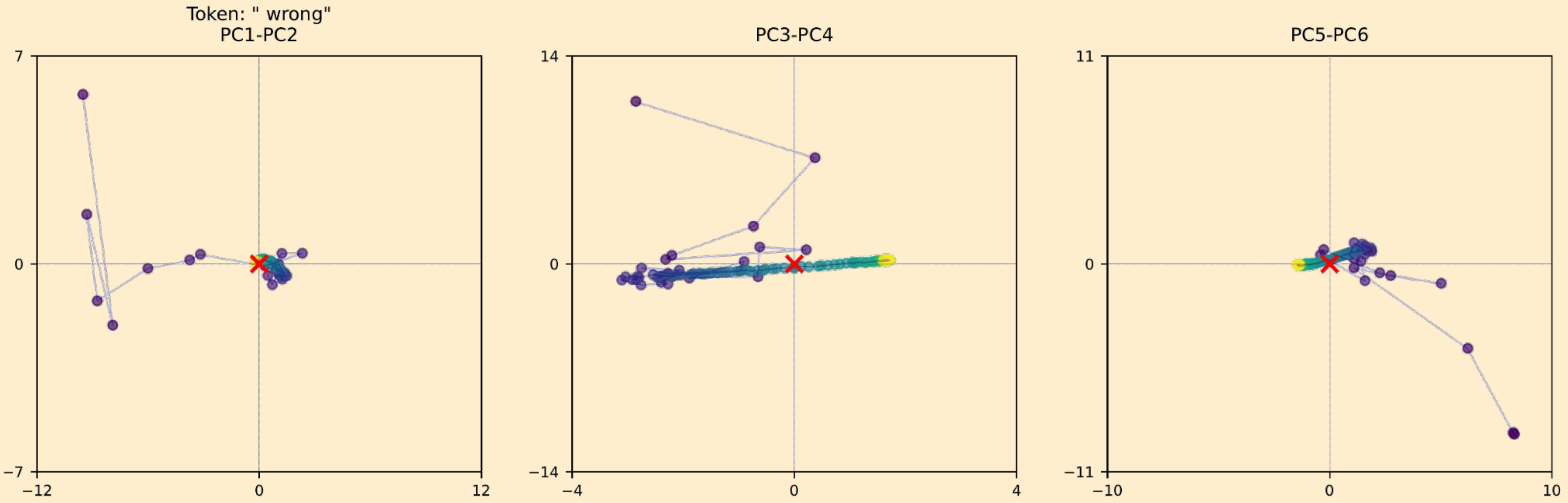
Emergent Terminal Behaviors



Emergent Terminal Behaviors



Emergent Terminal Behaviors



Takeaways from Trajectories

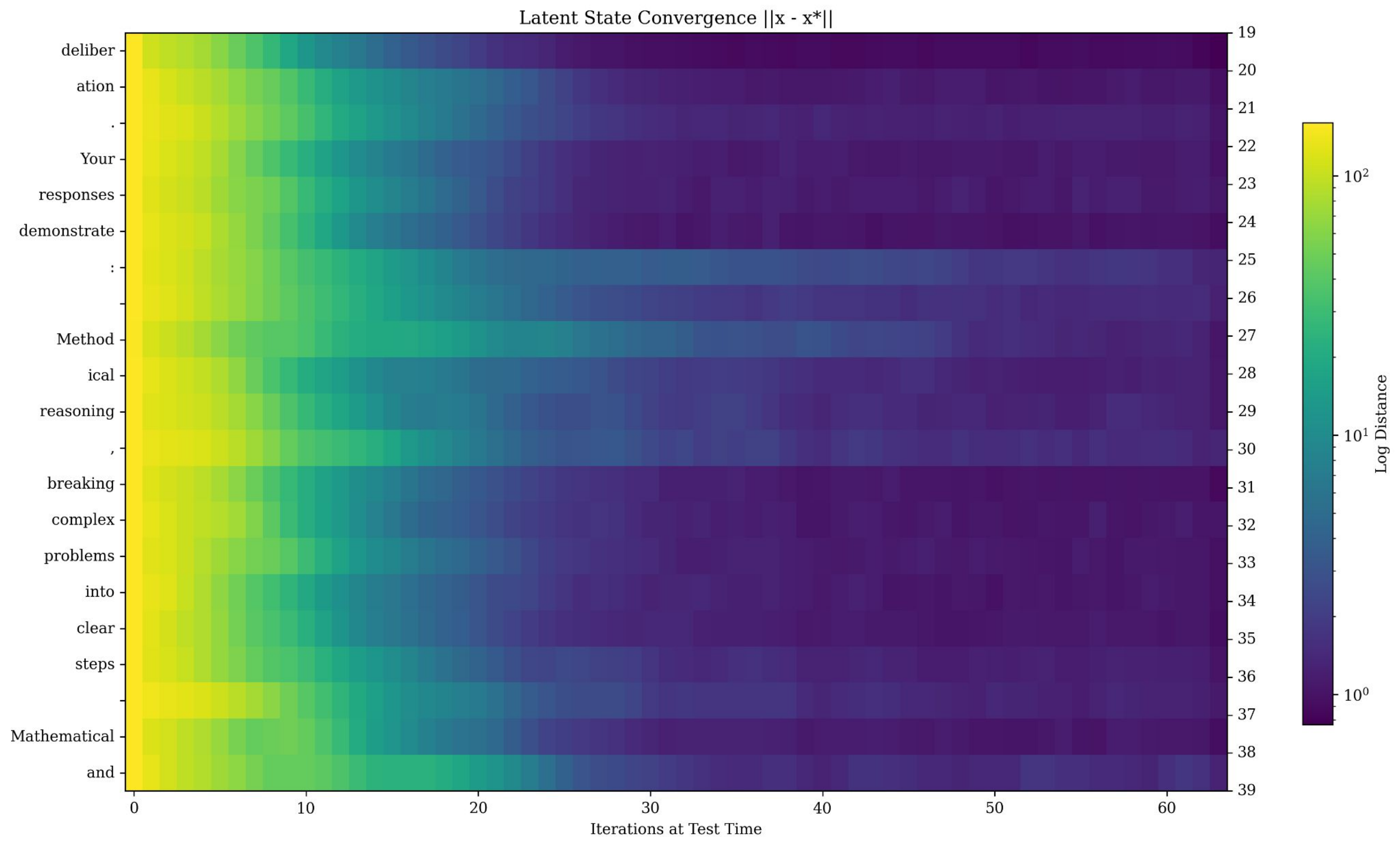
- Complexity emerges from pretraining
- Different terminal behaviors emerge from simple training objectives
- Harder to analyze model behavior -> requires representation analysis

Other Advantages of Recurrence

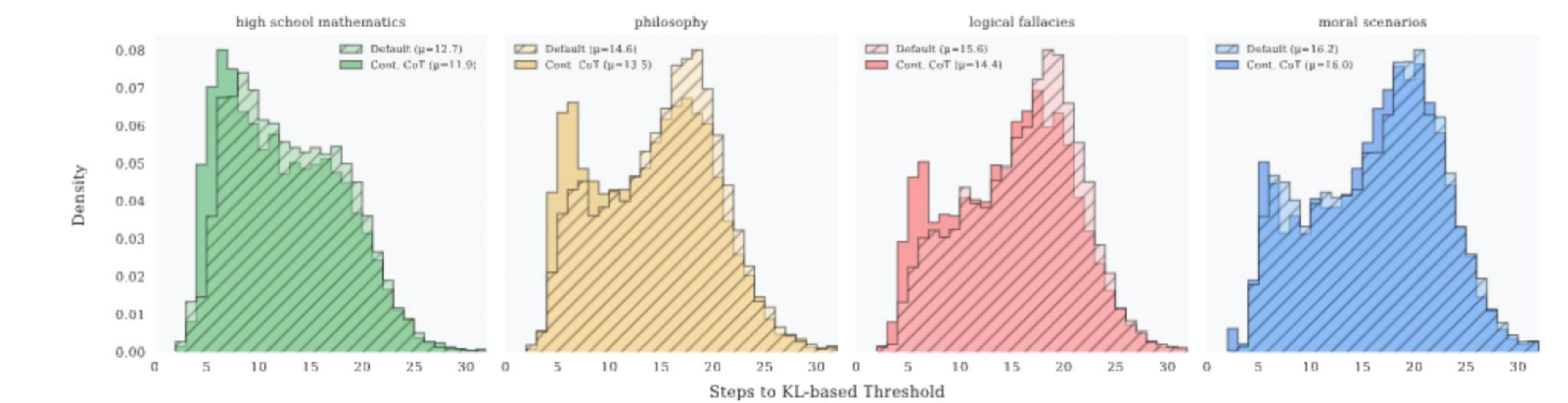
Recurrent Depth simplifies Speculative Decoding

Just run the model with a few recurrence steps to draft completions, verify with many steps

The model is able to exit *per-token* without training



The model is able to exit *per-token* without training



Recurrent Depth actually simplifies LLMs

- Simplified speculative decoding
- Zero-shot per-token adaptive computation
- Simplified KV-cache sharing
- Simplified continuous chain of thought

Conclusions, Takeaways, the Future

What would this do for recurrent neural nets?

- Efficient Sequence Mixers require depth to match attention
- Even a linear attention model could probe every token after (seq_length) many recurrence steps
- Would this be helpful?

If this is *compute-heavy, what is* parameter-heavy?

If this is *compute-heavy, what is* parameter-heavy?

Mixture-of-Experts Models :)

If this is *compute-heavy, what is* parameter-heavy?

Mixture-of-Experts Models :)

All types of "memory" implementations actually.
Which one is the best to combine with this?

Conclusions, Takeaways, the Future

- Different paradigm to pre-train models that scales surprisingly far
- How do we get arbitrary extrapolation in compute?
- How to post-train?
- Is this a complementary path to scaling model performance? What is an apples-to-apples comparison to CoT?

Questions