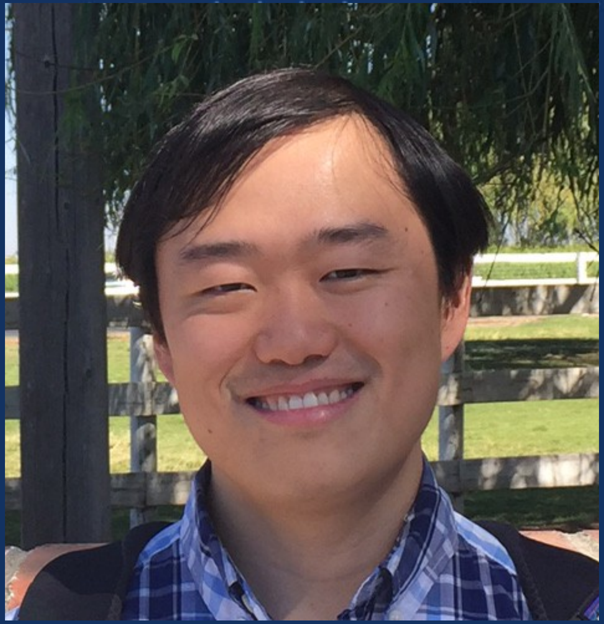
# Practical Algorithms from in-depth understanding of LLM behaviors

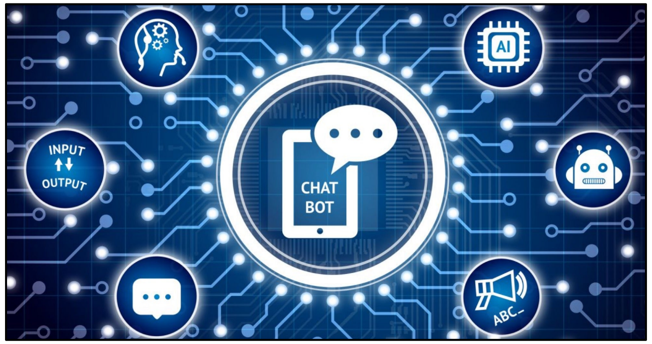
Yuandong Tian
Research Scientist Director

Meta Superintelligence Lab
(FAIR)



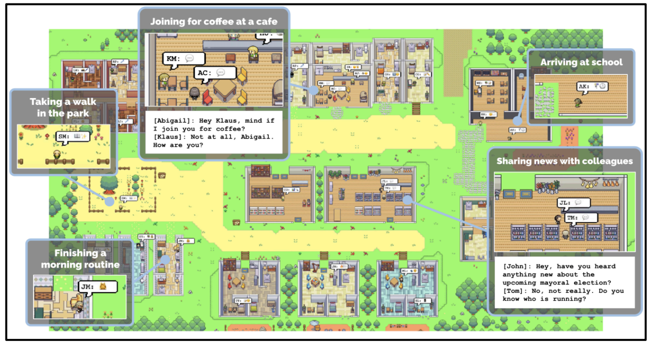
facebook Artificial Intelligence

# Large Language Models (LLMs)

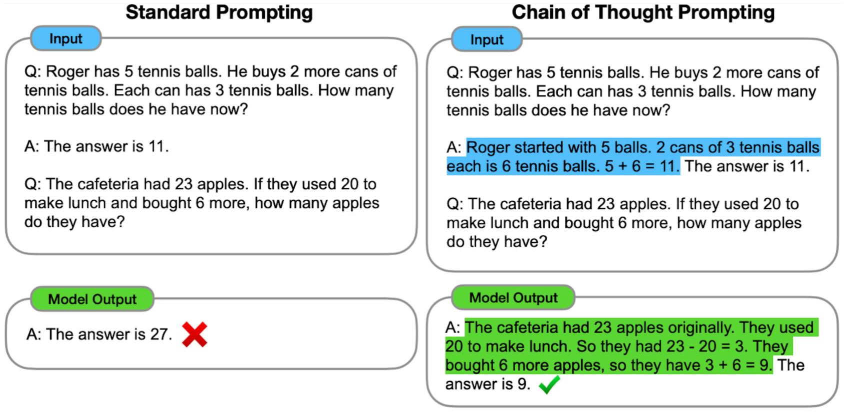



Conversational AI


Content Generation



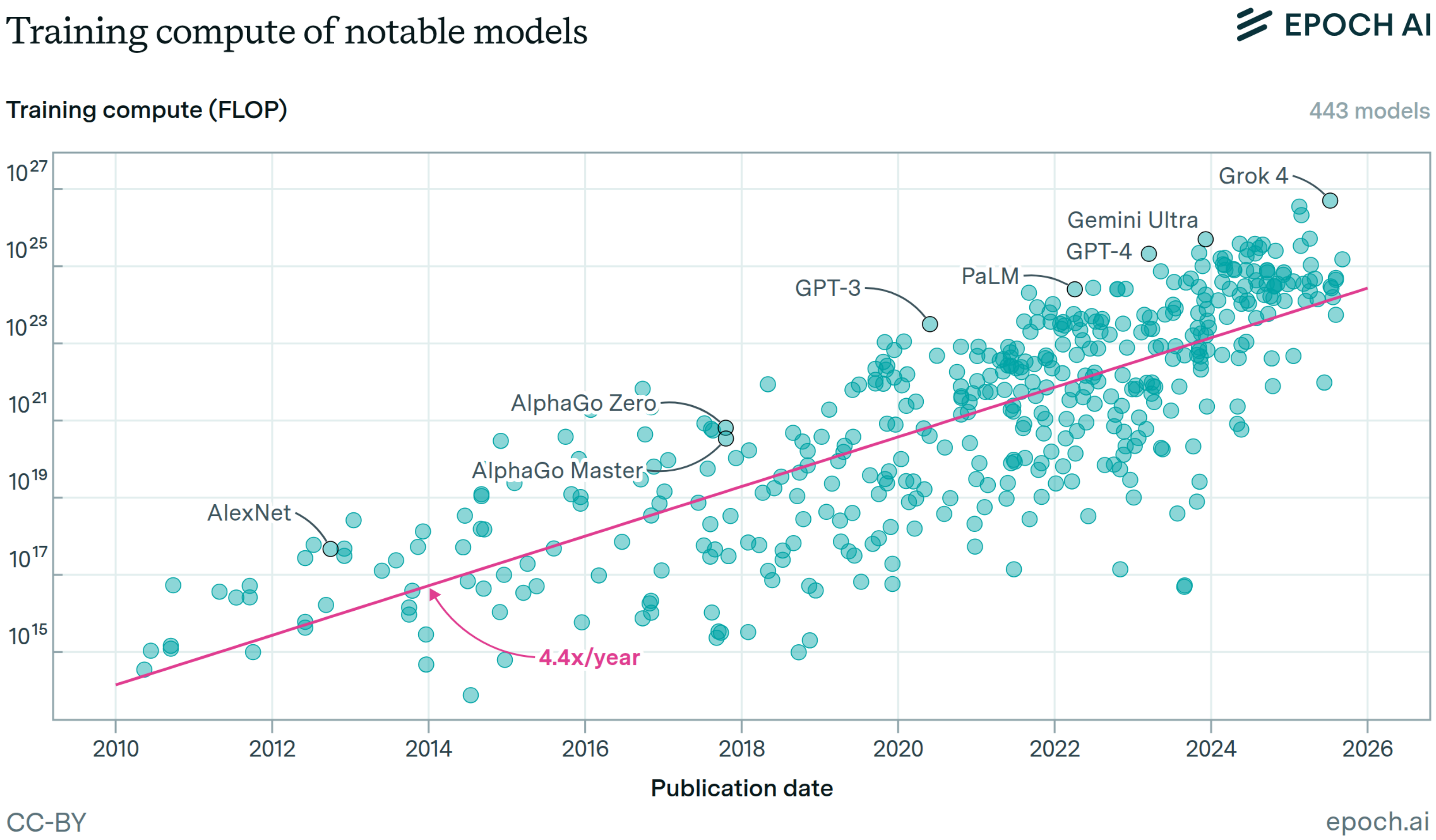

AI Agents



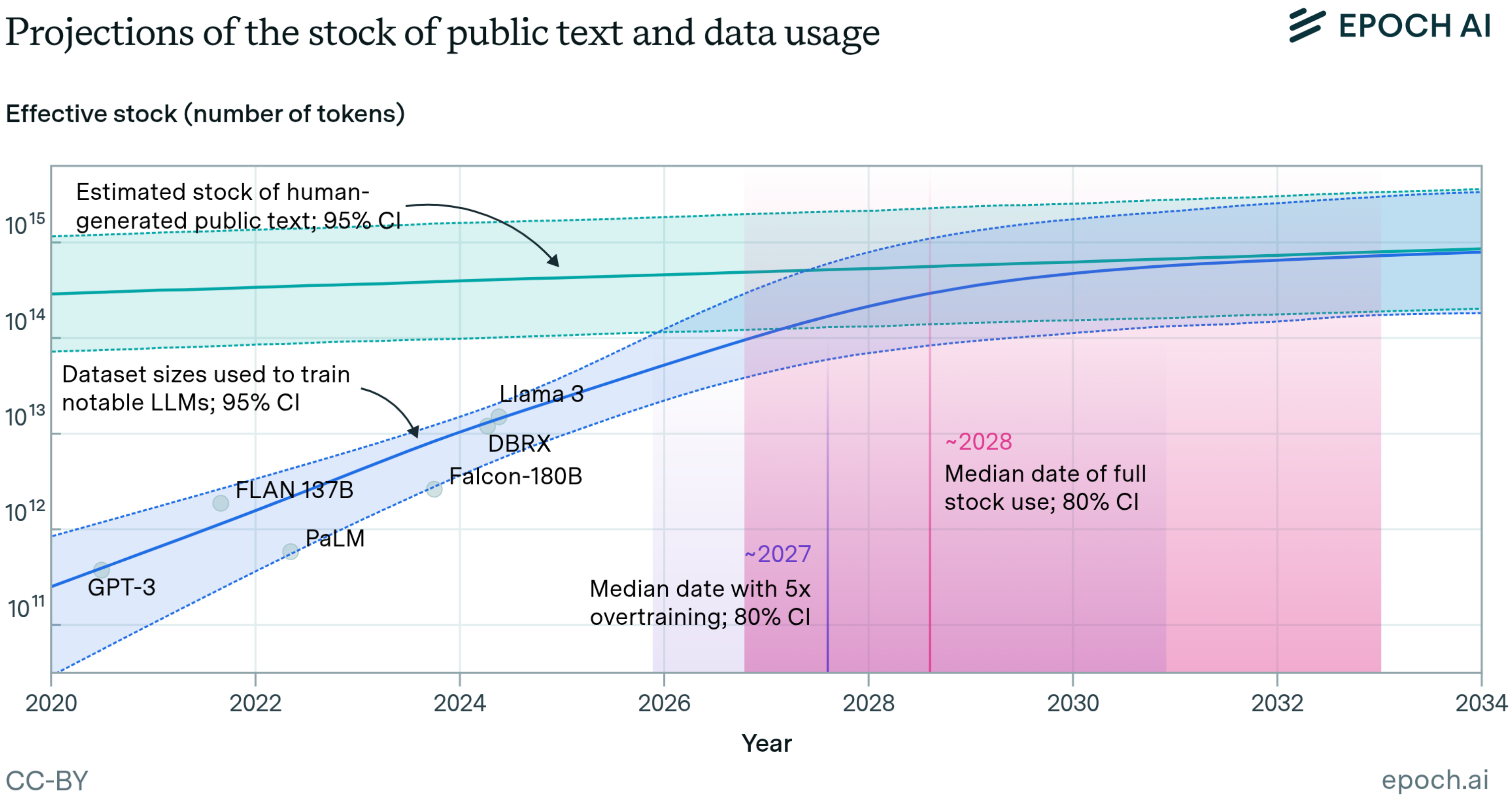

Reasoning


Planning

# The Progress of Large Models



# The Data Usage



# Comparison between Human and SoTA LMs

Question: Is our AI as strong as humans yet?

| | Training Data efficiency | Power Consumption | Adaptation to New Tasks | How to make decision? |
|---|---|---|---|---|
| **Human Brain** | < 10B text tokens, a lot of sensory inputs | Learning: ~20W<br>Inference/Thinking: ~20W | Learn with a few examples | By casual relationships and deep understanding |
| **Sota LMs** | ~10T-50T tokens | Learning: at least @ MWh<br>Inference/Thinking: 1W-30W | Hundreds / Thousands of data points. May fail to generalize | Correlation & Pattern Matching |

**Estimated #tokens consumed by human in the life time:** 70 years * 300 days / year * 12 hours / day * 3600 seconds / hours * 10 tokens / second = 9.1B

**Andrej Karpathy**
@karpathy

My pleasure to come on Dwarkesh last week, I thought the questions and conversation were really good.

I re-watched the pod just now too. First of all, yes I know, and I'm sorry that I speak so fast :). It's to my detriment because sometimes my speaking thread out-executes my thinking thread, so I think I botched a few explanations due to that, and sometimes I was also nervous that I'm going too much on a tangent or too deep into something relatively spurious. Anyway, a few notes/pointers:

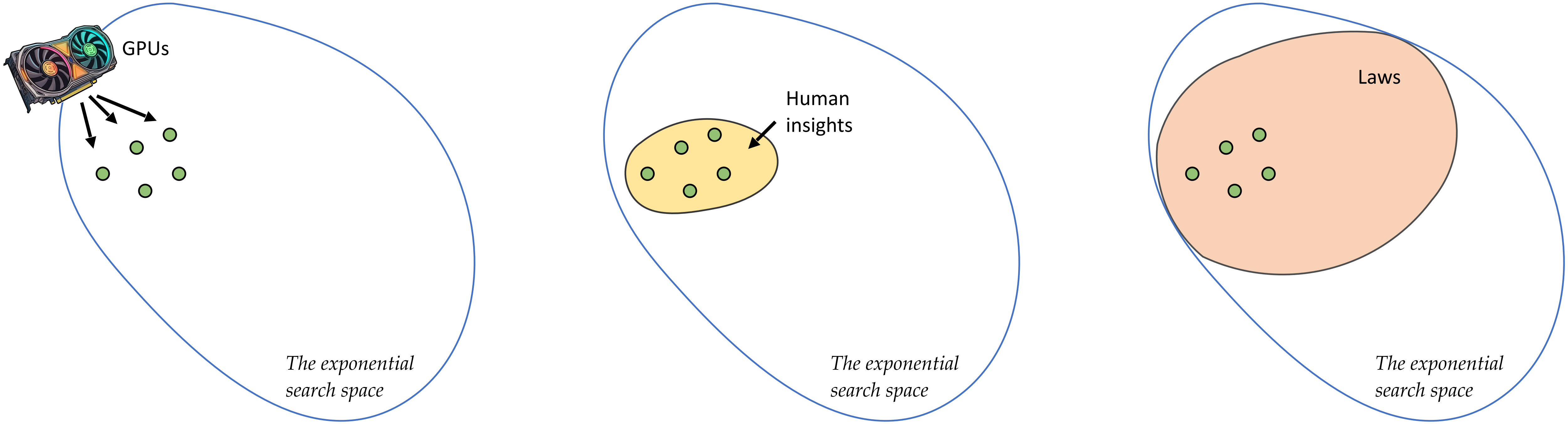
**AGI timelines.** My comments on AGI timelines looks to be the most trending part of the early response. This is the "decade of agents" is a reference to this earlier tweet x.com/karpathy/statu... Basically my AI timelines are about 5-10X pessimistic w.r.t. what you'll find in your neighborhood SF AI house party or on your twitter timeline, but still quite optimistic w.r.t. a rising tide of AI deniers and skeptics. The apparent conflict is not: imo we simultaneously 1) saw a huge amount of progress in recent years with LLMs while 2) there is still a lot of work remaining (grunt work, integration work, sensors and actuators to the physical world, societal work, safety and security work (jailbreaks, poisoning, etc.)) and also research to get done before we have an entity that you'd prefer to hire over a person for an arbitrary job in the world. I think that overall, 10 years should otherwise be a very bullish timeline for AGI, it's only in contrast to present hype that it doesn't feel that way.

# How we should do our research from now on?

- The “Data Wall problem”
  - We may have used all the available data on the Internet.
  - How to deal with corner cases / personalization / private data?
  - Human is still much more efficient than current AI

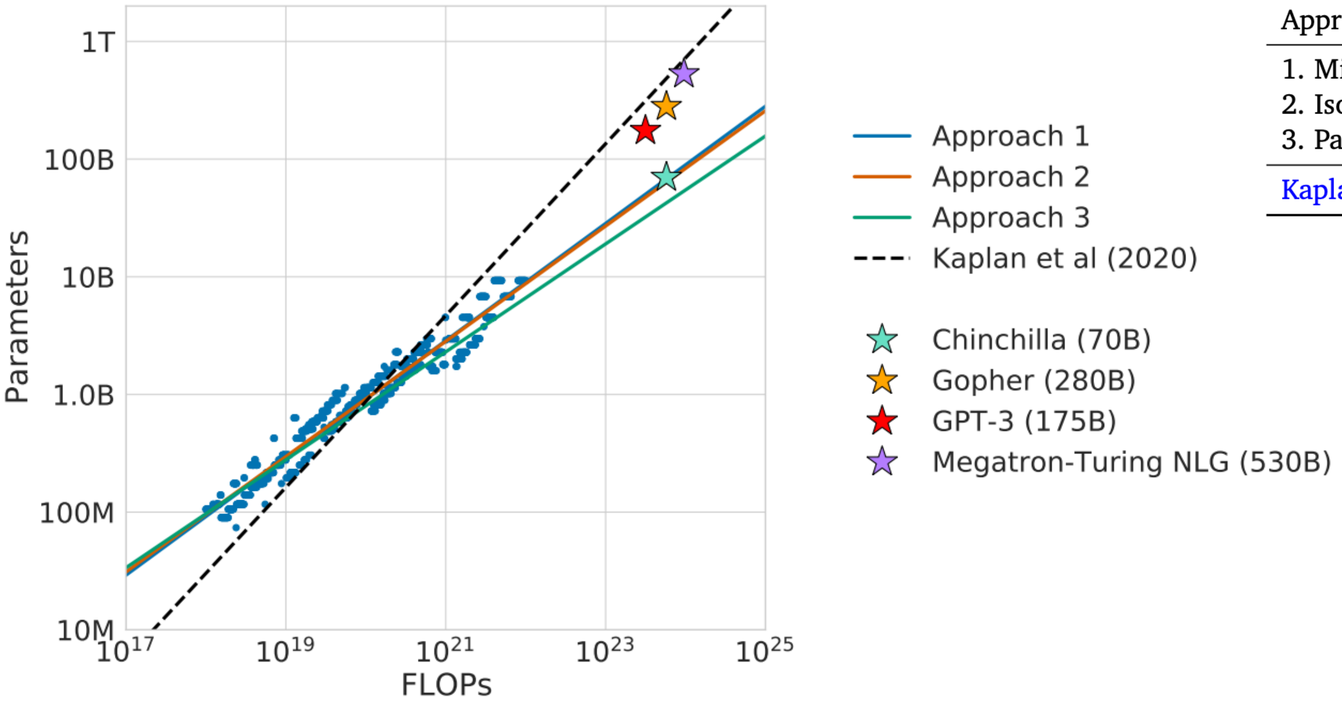
- Everyone is GPU poor
  - What are new axes to scale? GPUs are never enough.
  - Data itself cannot extrapolate, only human insights can.

# The New (a.k.a. Old) Scaling Axis




Question: Can we **scale** the scaling laws?

# How we get Scaling Laws?




| Approach | Coeff. $a$ where $N_{opt} \propto C^a$ | Coeff. $b$ where $D_{opt} \propto C^b$ |
|---|---|---|
| 1. Minimum over training curves | 0.50 (0.488, 0.502) | 0.50 (0.501, 0.512) |
| 2. IsoFLOP profiles | 0.49 (0.462, 0.534) | 0.51 (0.483, 0.529) |
| 3. Parametric modelling of the loss | 0.46 (0.454, 0.455) | 0.54 (0.542, 0.543) |
| Kaplan et al. (2020) | 0.73 | 0.27 |

Steps:

1. Collect the experiments
2. Form hypothesis (linear, power-law, etc)
3. Extrapolate

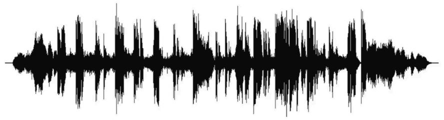
Still pure statistics and need exponential data.

(No leverage of the knowledge of architecture/data)

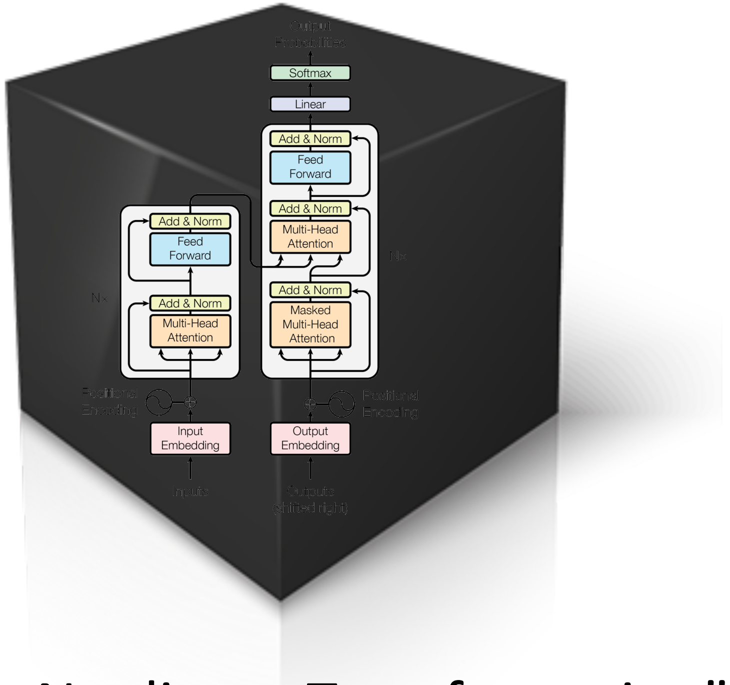
# How does deep learning work?

Input

Output

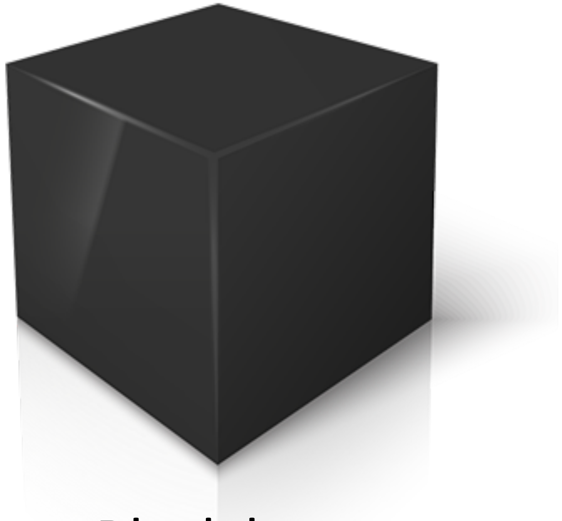


This is an apple

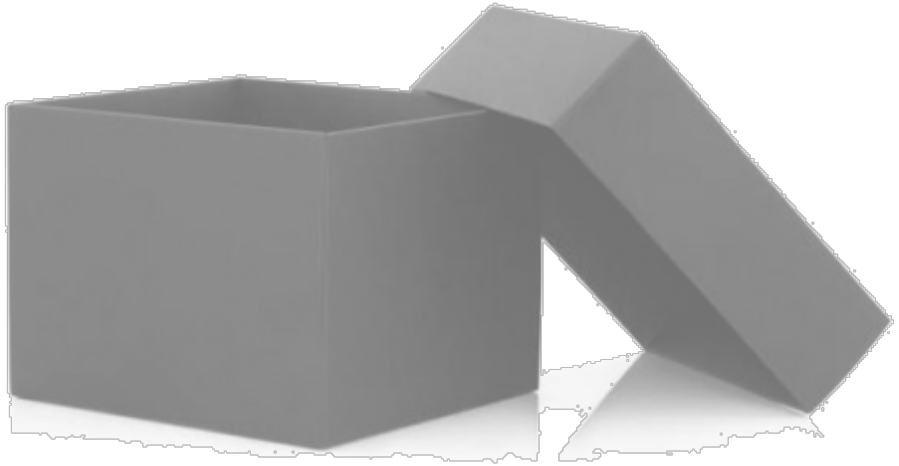



“Some Nonlinear Transformation”

# Black-box versus White-box



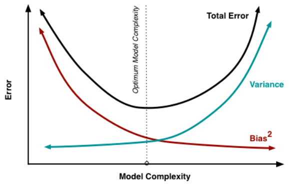
Black box



White box

# What routes should we take?

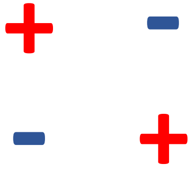
**Generalization**




Architecture ✗

training dynamics ✗

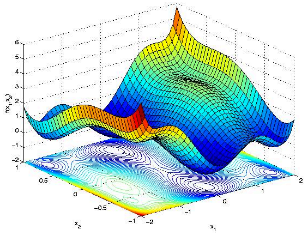
**Expressibility**



Architecture ✓

training dynamics ✗

**Optimization**



Architecture ✗

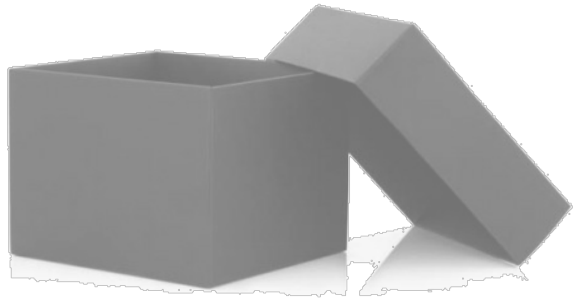
training dynamics ✓

**How about**

Architecture ✓

training dynamics ✓

# Start From the First Principle



- Training follows Gradient and its variants (SGD, Adams, etc)

$$\dot{\boldsymbol{w}} := \frac{\mathrm{d}\boldsymbol{w}}{\mathrm{d}t} = -\nabla_{\boldsymbol{w}} J(\boldsymbol{w})$$

- **First principle** → Understand the behavior of the neural networks by checking the gradient **dynamics** induced by the neural **architectures**.

- Sounds complicated.. Is that possible? **Yes**

Architecture ✓

training dynamics ✓

# What Gradient Descent gives us?

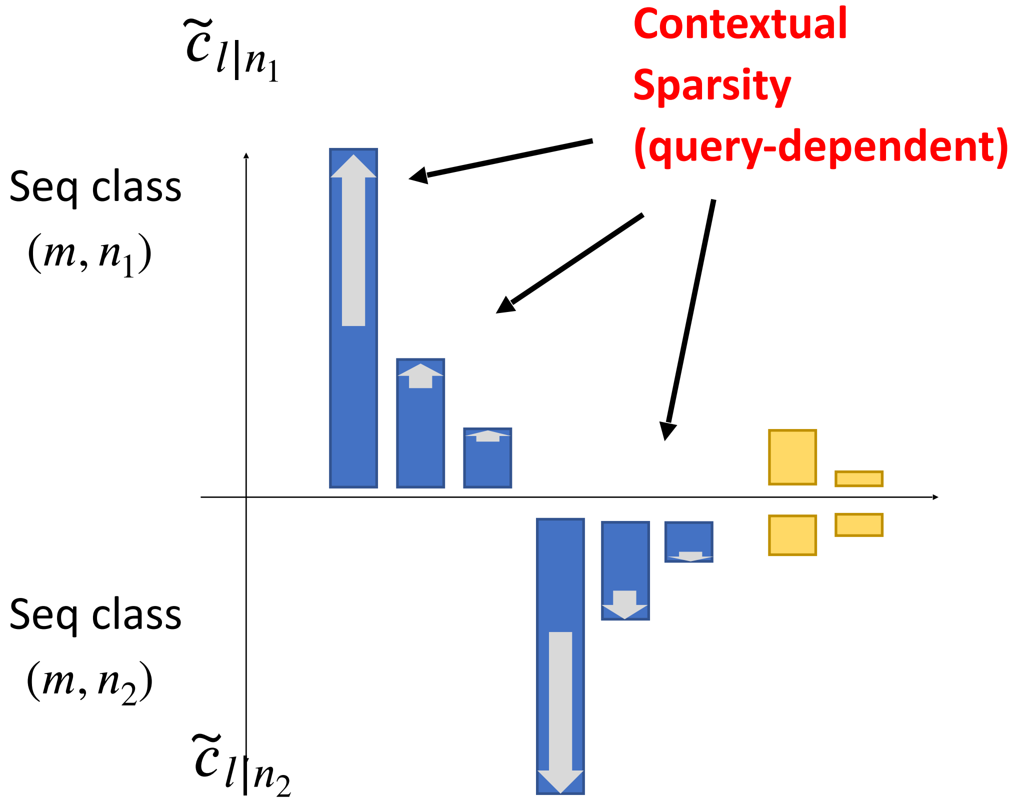
**Short-term:**
Finding Simple Structures
(Low-rank, sparsity)

**Long-term:**
How the representation is learned
(Key to the success of deep models)

Leverage Them in Practical Algorithms

# **Short-term:** Finding Nice Structures

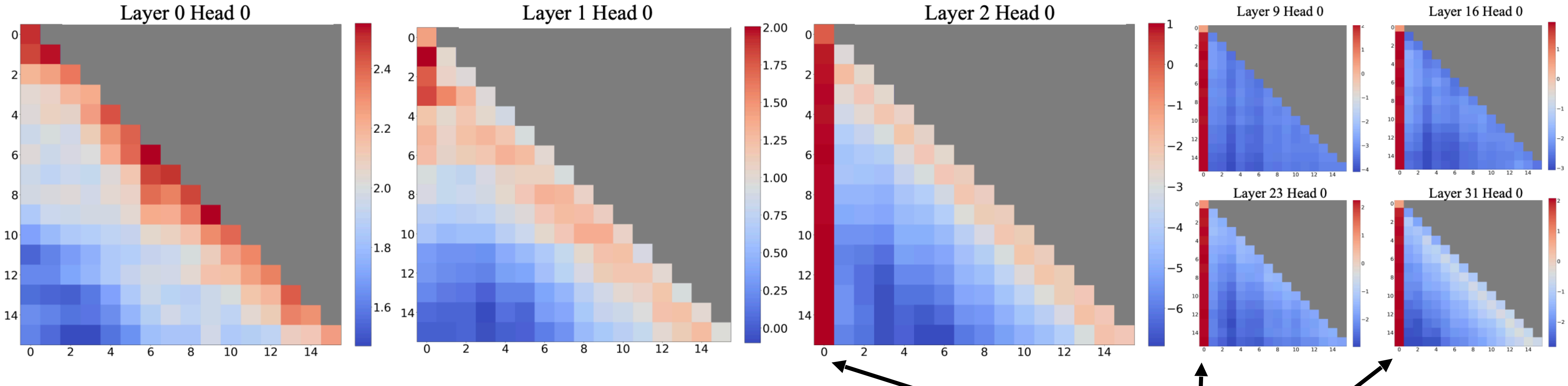
# Attention Sparsity




***Attention = Learnable*** TF-IDF (Term Frequency, Inverse Document Frequency)

[*Y. Tian et al*, Scan and Snap: Understanding Training Dynamics and Token Composition in 1-layer Transformer, *NeurIPS'23*]

# Attention Sinks: Initial tokens draw a lot of attentions




Average attention logits in Llama-2-7B over 256 sentences

First few tokens!!

- Observation: Initial tokens have large attention scores, even if they're not semantically significant.
- Attention Sink: Tokens that disproportionately attract attention irrespective of their relevance.



*[G. Xiao et al, Efficient Streaming Language Models with Attention Sinks, ICLR'24]*

# Understanding Attention Sinks

- **Why?** Attention scores **have to** sum up to **1** for all contextual tokens. (*SoftMax-Off-by-One, Miller et al. 2023*)

$$\mathrm{SoftMax}(x)_i = \frac{e^{x_i}}{e^{x_1} + \sum_{j=2}^{N} e^{x_j}}, \quad x_1 \gg x_j, j \in 2, \ldots, N$$

- **Why initial tokens?** Their visibility to subsequent tokens, rooted in autoregressive language modeling.

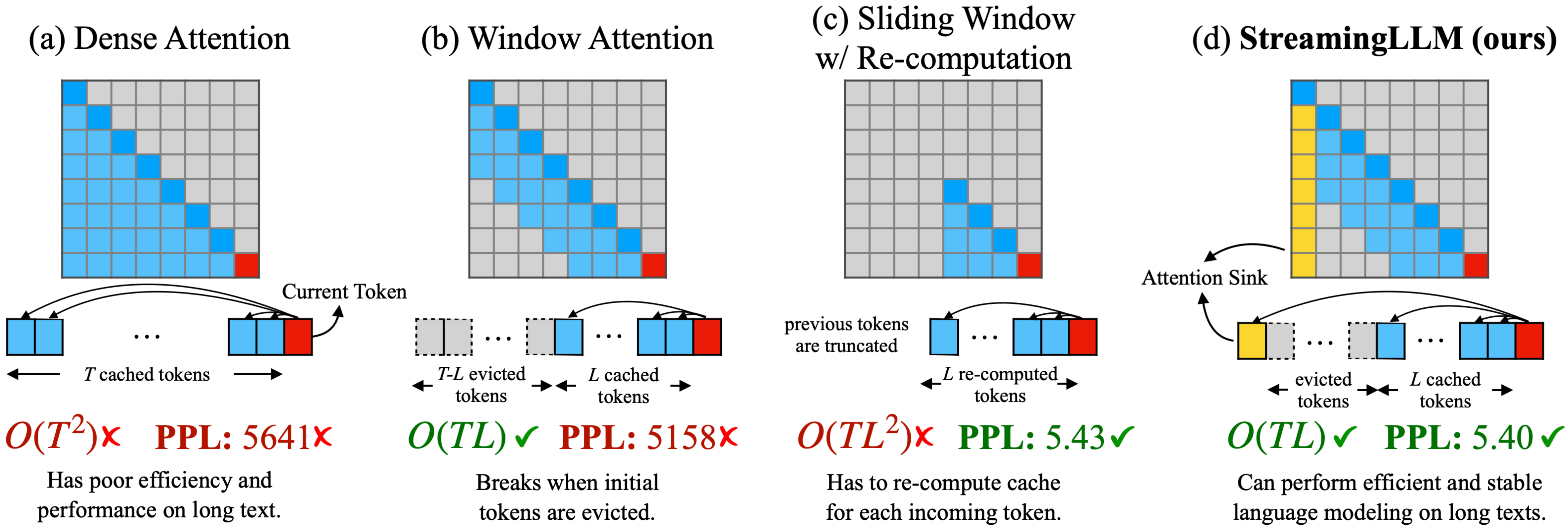
- The model learns a bias towards their **absolute position** rather than the semantics.

| Llama-2-13B | PPL (↓) |
| --- | --- |
| 0+1024 (window) | 5158.07 |
| 4+1024 | 5.40 |
| 4”\n”+1020 | 5.6 |



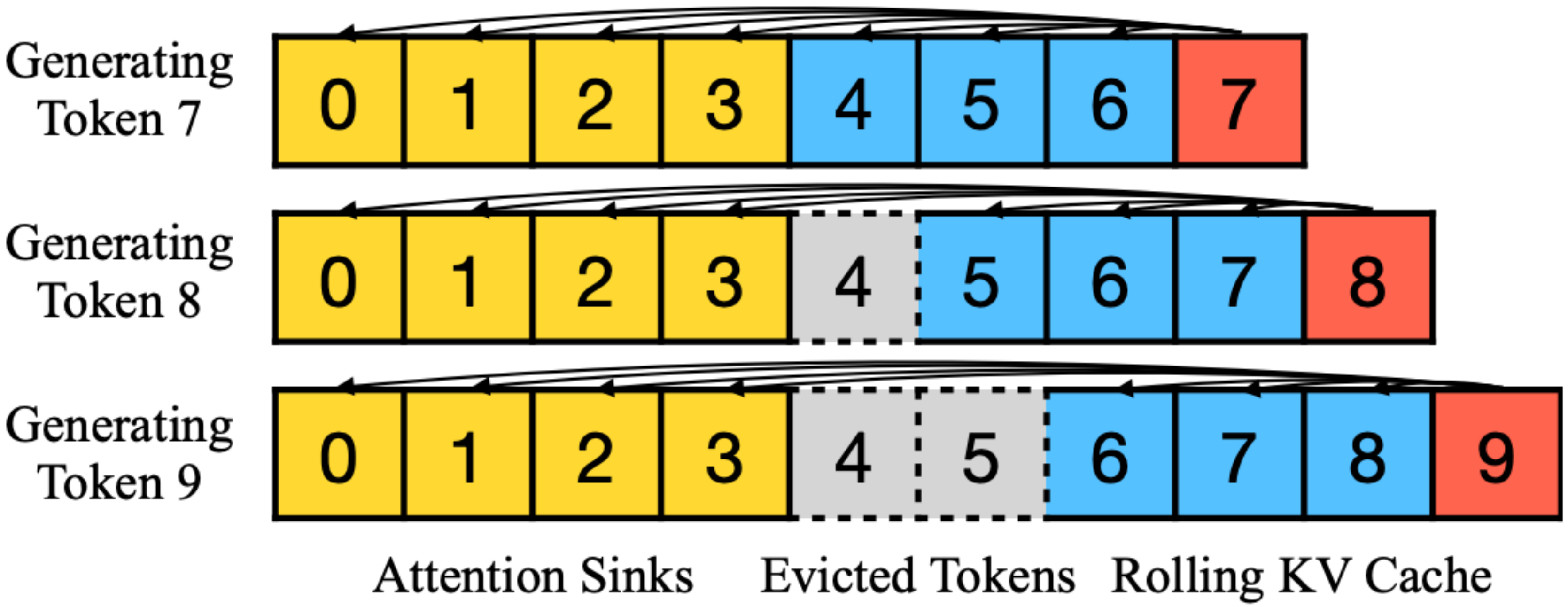
*[G. Xiao et al, Efficient Streaming Language Models with Attention Sinks, ICLR’24]*

# StreamingLLM




*[G. Xiao et al, Efficient Streaming Language Models with Attention Sinks, ICLR'24]*

# StreamingLLM

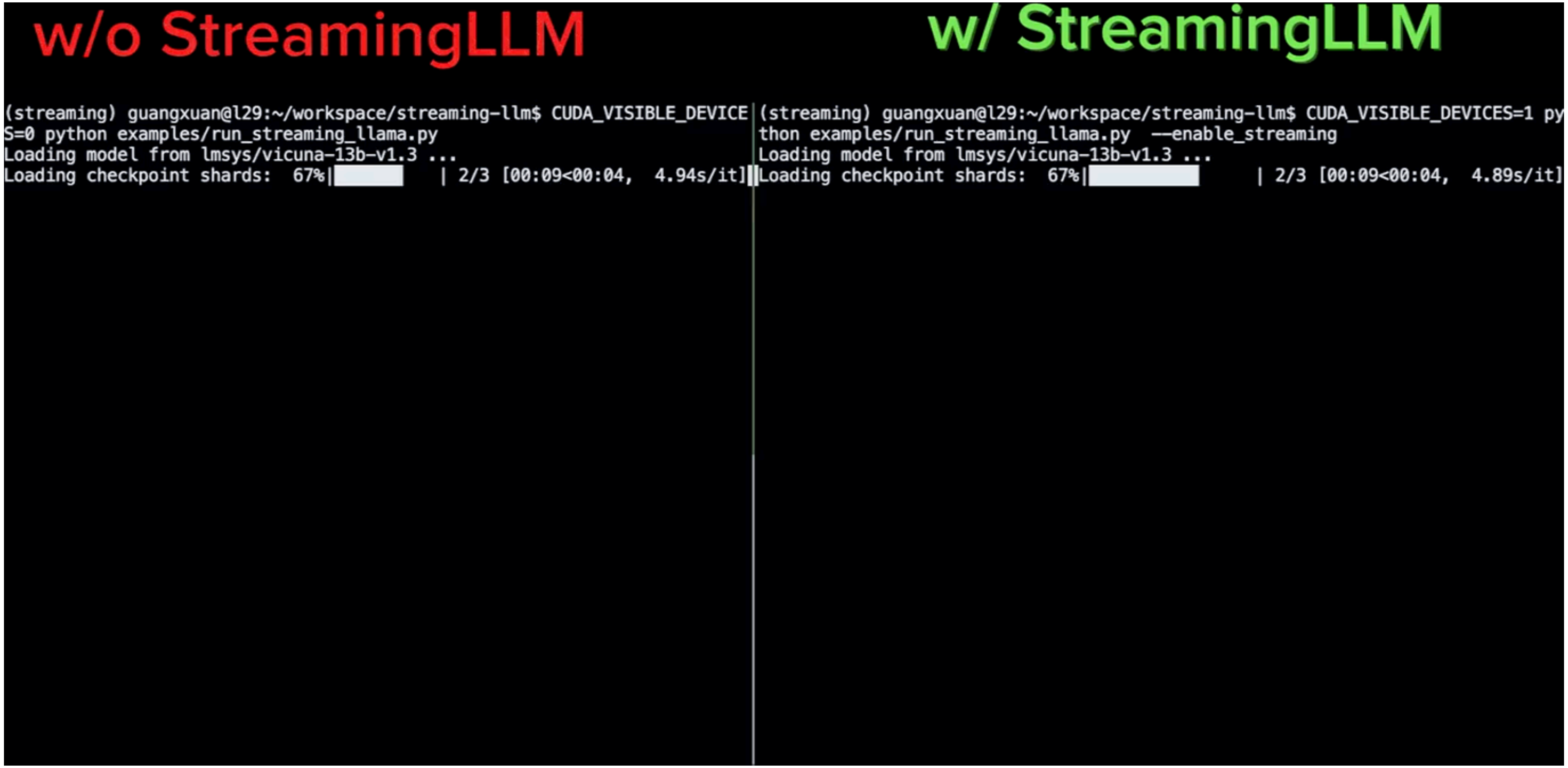



**Key design: Position Rolling**

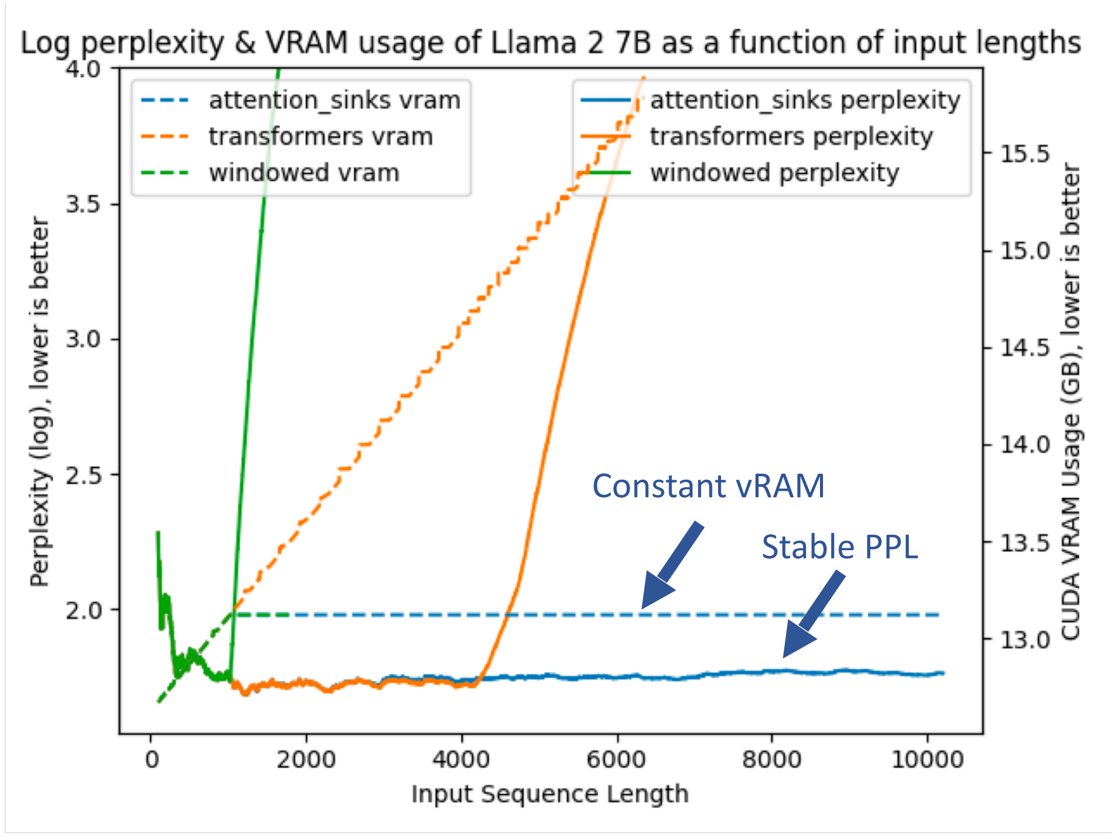
For all tokens, use their positions **within cache** to compute positional encoding!

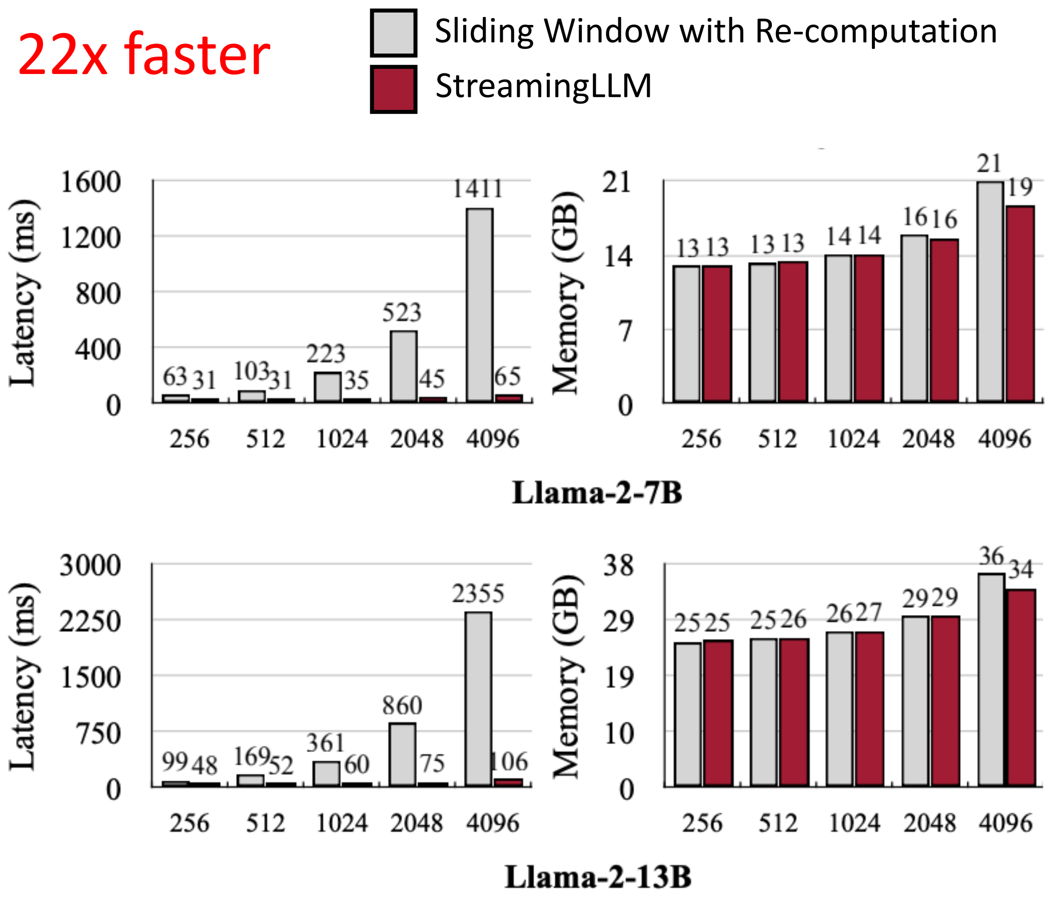
→ Token distance never exceeds pre-trained context window!

# StreamingLLM



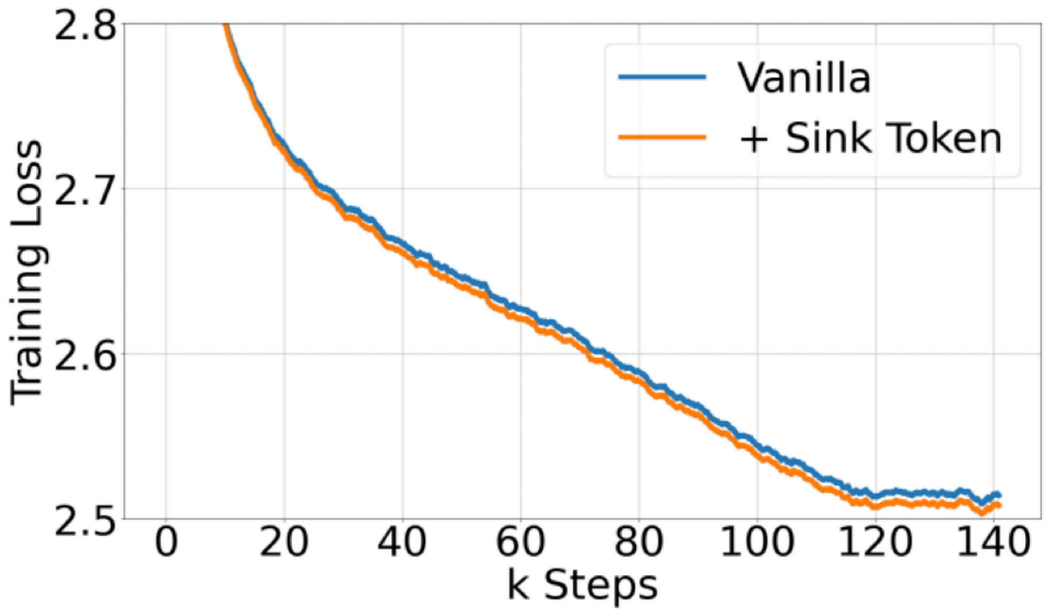
# StreamingLLM: stable PPL, constant vRAM



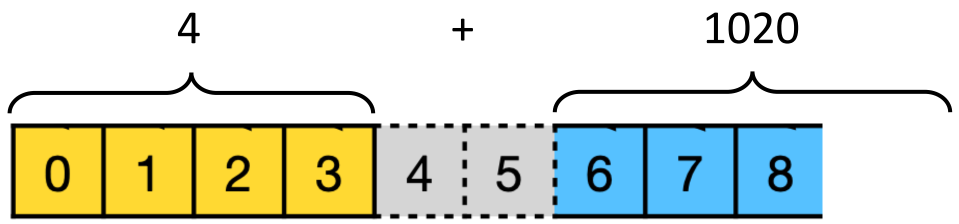



# Understanding Attention Sinks

- Pre-train with a Dedicated Attention Sink Token




| Cache Config | PPL (↓) | | | |
|---|---|---|---|---|
| | 0+1024 | 1+1023 | 2+1022 | 4+1020 |
| Vanilla | 27.87 | 18.49 | 18.05 | 18.05 |
| Zero Sink | 29214 | 19.90 | 18.27 | 18.01 |
| Learnable Sink | 1235 | 18.01 | 18.01 | 18.02 |

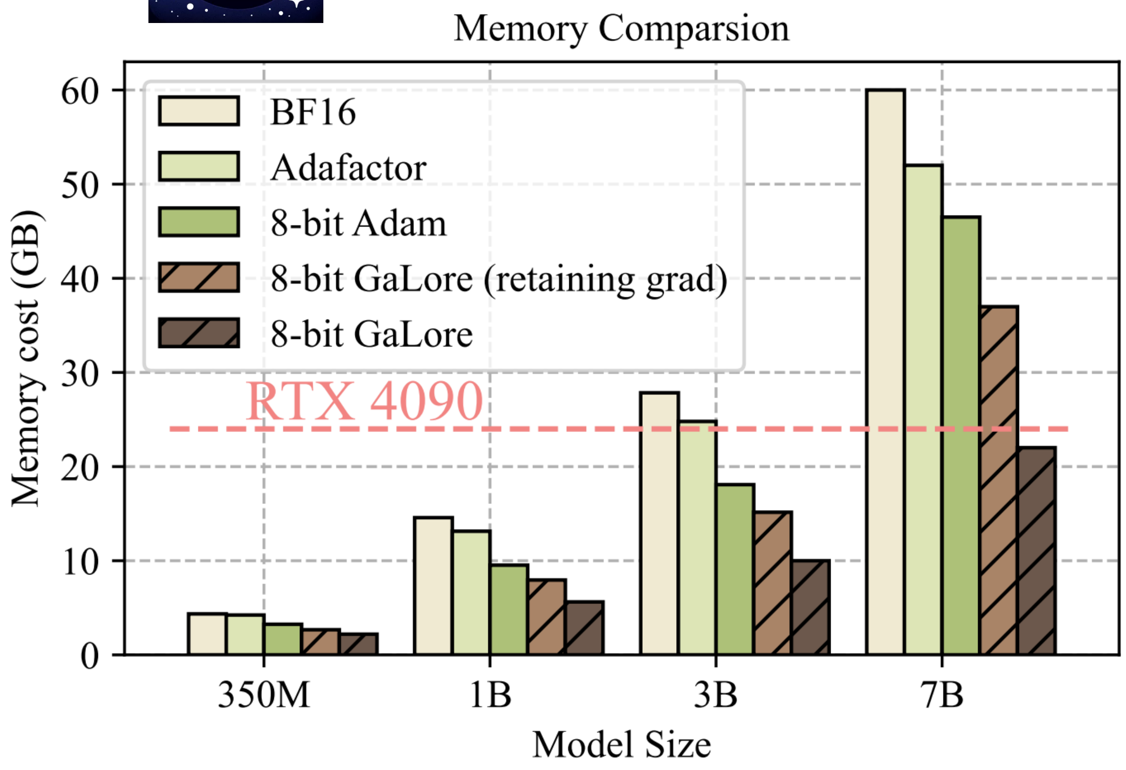


# Impact

**Attention:** Following GPT-3, attention blocks alternate between banded window and fully dense patterns [8][9], where the bandwidth is 128 tokens. Each layer has 64 query heads of dimension 64, and uses Grouped Query Attention (GQA [10][11]) with 8 key-value heads. We apply rotary position embeddings [12] and extend the context length of dense layers to 131,072 tokens using YaRN [13]. Each attention head has a learned bias in the denominator of the softmax, similar to off-by-one attention and attention sinks [14][15], which enables the attention mechanism to pay no attention to any tokens.

- 900+ citations
- Used in GPT OSS models in pre-training

# GaLore: Pre-training 7B model on RTX 4090 (24G)




| | Rank | Retain grad | Memory | Token/s |
|---|---|---|---|---|
| 8-bit AdamW | | Yes | 40GB | 1434 |
| 8-bit GaLore | 16 | Yes | 28GB | 1532 |
| 8-bit GaLore | 128 | Yes | 29GB | 1532 |
| 16-bit GaLore | 128 | Yes | 30GB | **1615** |
| 16-bit GaLore | 128 | No | **18GB** | 1587 |
| 8-bit GaLore | 1024 | Yes | 36GB | 1238 |

* SVD takes around 10min for 7B model, but runs every T=500-1000 steps.

Third-party evaluation by @llamafactory_ai



[J. Zhao et al, *GaLore: Memory-Efficient LLM Training by Gradient Low-Rank Projection*, ICML'24 (Oral)]

# Memory Saving with GaLore

**Algorithm 1:** GaLore, PyTorch-like

```
for weight in model.parameters():
    grad = weight.grad
    # original space -> compact space
    lor_grad = project(grad)
    # update by Adam, Adafactor, etc.
    lor_update = update(lor_grad)
    # compact space -> original space
    update = project_back(lor_update)
    weight.data += update
```

## GaLore

$G_t \leftarrow -\nabla_W \phi(W_t)$

If t % T == 0:

Compute $P_t = \text{SVD}(G_t) \in \mathbb{R}^{m \times r}$

$R_t \leftarrow P_t^T G_t$ {project}

$\tilde{R}_t \leftarrow \rho(R_t)$ {Adam in low-rank}

$\tilde{G}_t \leftarrow P_t \tilde{R}_t$ {project-back}

$W_{t+1} \leftarrow W_t + \eta \tilde{G}_t$

| Memory Usage | Weight | Optim States | Projection | Total |
|---|---|---|---|---|
| Full-rank | $mn$ | $2mn$ | 0 | $3mn$ |
| Low-rank adaptor | $mn+mr+nr$ | $2mr+2nr$ | 0 | $2mn+3mr+3nr$ |
| GaLore | $mn$ | $mr+2nr$ | $mr$ | $mn+2mr+2nr$ |

$W_t$ $R_t$ $P_t$

# Pre-training Results (LLaMA 7B)

| Params | Hidden | Intermediate | Heads | Layers | Steps | Data amount |
|---|---|---|---|---|---|---|
| 60M | 512 | 1376 | 8 | 8 | 10K | 1.3 B |
| 130M | 768 | 2048 | 12 | 12 | 20K | 2.6 B |
| 350M | 1024 | 2736 | 16 | 24 | 60K | 7.8 B |
| 1 B | 2048 | 5461 | 24 | 32 | 100K | 13.1 B |
| 7 B | 4096 | 11008 | 32 | 32 | 150K | 19.7 B |

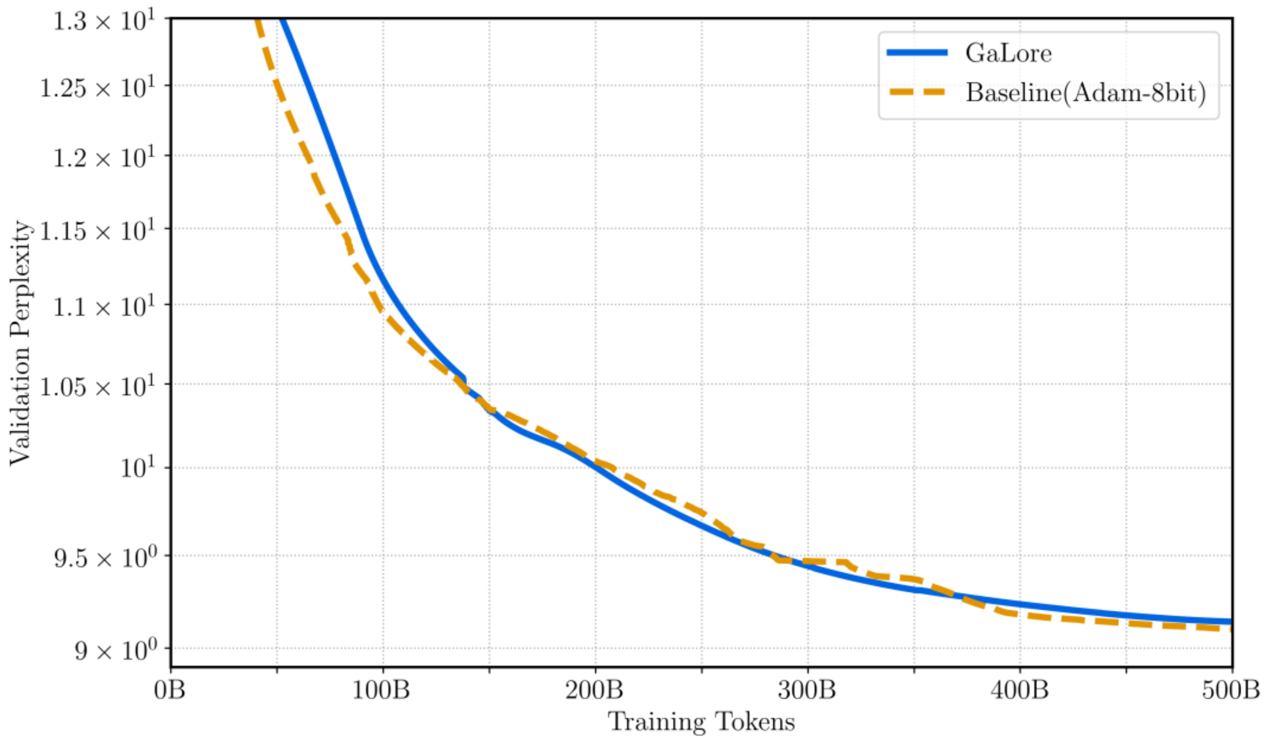
| | Mem | 40K | 80K | 120K | 150K |
|---|---|---|---|---|---|
| **8-bit GaLore** | 18G | 17.94 | 15.39 | 14.95 | 14.65 |
| 8-bit Adam | 26G | 18.09 | 15.47 | 14.83 | 14.61 |
| Tokens (B) | | 5.2 | 10.5 | 15.7 | 19.7 |

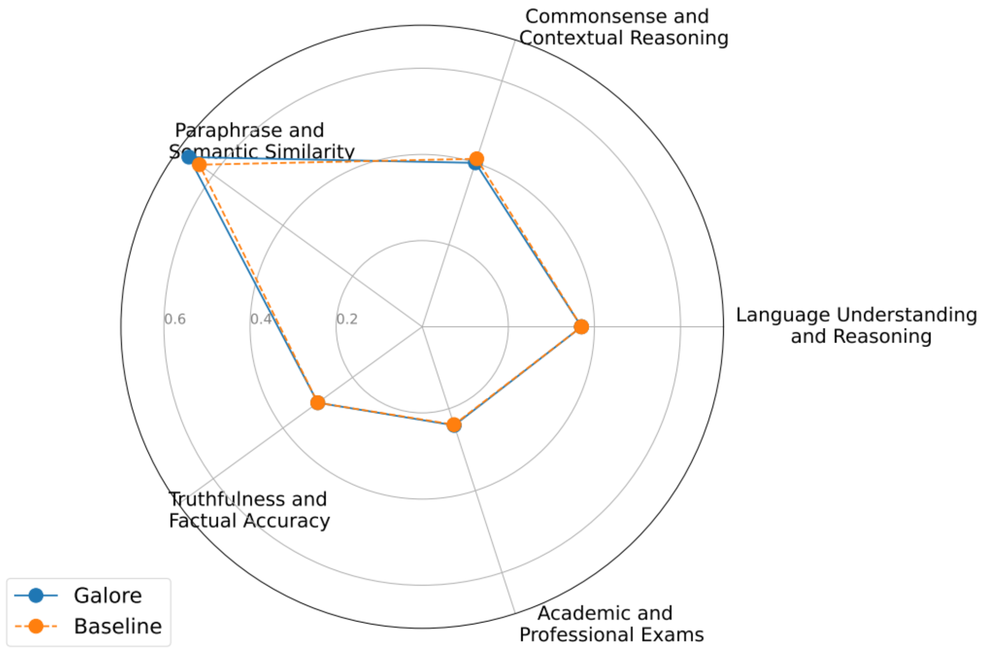
* Experiments are conducted on 8 x 8 A100

| | **60M** | **130M** | **350M** | **1B** |
|---|---|---|---|---|
| Full-Rank | 34.06 (0.36G) | 25.08 (0.76G) | 18.80 (2.06G) | 15.56 (7.80G) |
| **GaLore** | **34.88** (0.24G) | **25.36** (0.52G) | **18.95** (1.22G) | **15.64** (4.38G) |
| Low-Rank | 78.18 (0.26G) | 45.51 (0.54G) | 37.41 (1.08G) | 142.53 (3.57G) |
| LoRA | 34.99 (0.36G) | 33.92 (0.80G) | 25.58 (1.76G) | 19.21 (6.17G) |
| ReLoRA | 37.04 (0.36G) | 29.37 (0.80G) | 29.08 (1.76G) | 18.33 (6.17G) |
| $r/d_{model}$ | 128 / 256 | 256 / 768 | 256 / 1024 | 512 / 2048 |
| Training Tokens | 1.1B | 2.2B | 6.4B | 13.1B |

* On LLaMA 1B, ppl is better (~14.97) with ½ rank (1024/2048)

# Pre-training Results (LLaMA 7B)






[D. Su et al, *GaLore 2: Large-Scale LLM Pre-Training by Gradient Low-Rank Projection*]

**Long-term:** How Network finds Representation

# Type of Representations

Representation

(Traditional) Symbolic representation

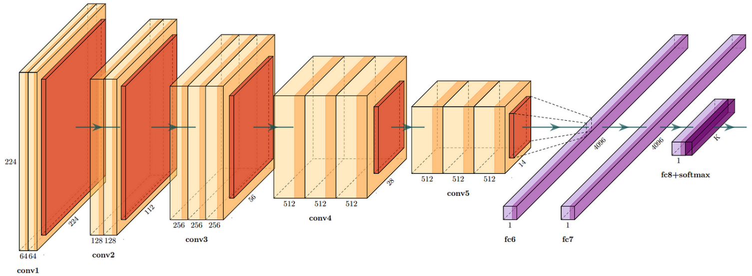
$$\nabla \cdot \mathbf{E} = \frac{\rho_v}{\varepsilon} \quad \text{(Gauss' Law)}$$

$$\nabla \cdot \mathbf{H} = 0 \quad \text{(Gauss' Law for Magnetism)}$$

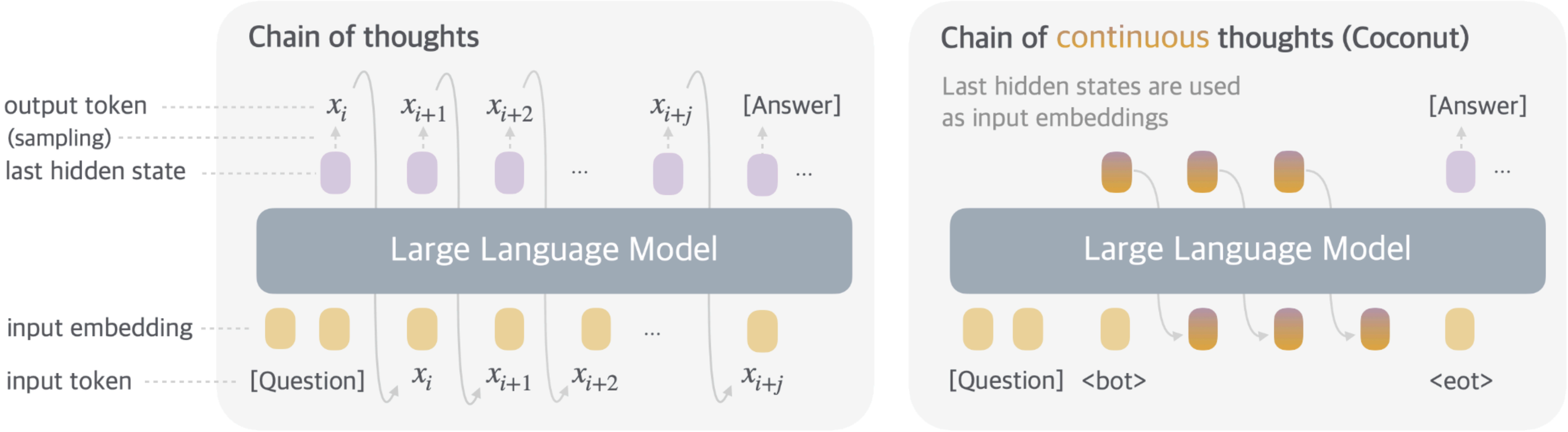
$$\nabla \times \mathbf{E} = -\mu \frac{\partial \mathbf{H}}{\partial t} \quad \text{(Faraday's Law)}$$

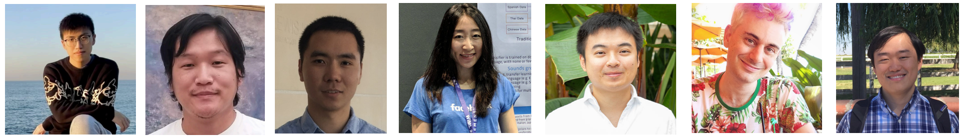
$$\nabla \times \mathbf{H} = \mathbf{J} + \varepsilon \frac{\partial \mathbf{E}}{\partial t} \quad \text{(Ampere's Law)}$$

Neural Representation



# CoConut (Chain of Continuous Thought)



[S. Hao et al, *Training Large Language Models to Reason in a Continuous Latent Space*, COLM'25]

# How to train Coconut?

**Language CoT**
*(training data)*

[Question] [Step 1] [Step 2] [Step 3] ··· [Step N] [Answer]

[Thought] : continuous thought
[ ··· ] : sequence of tokens
<···> : special token
··· : calculating loss

**Stage 0** [Question] <bot> <eot> [Step 1] [Step 2] ··· [Step N] [Answer]

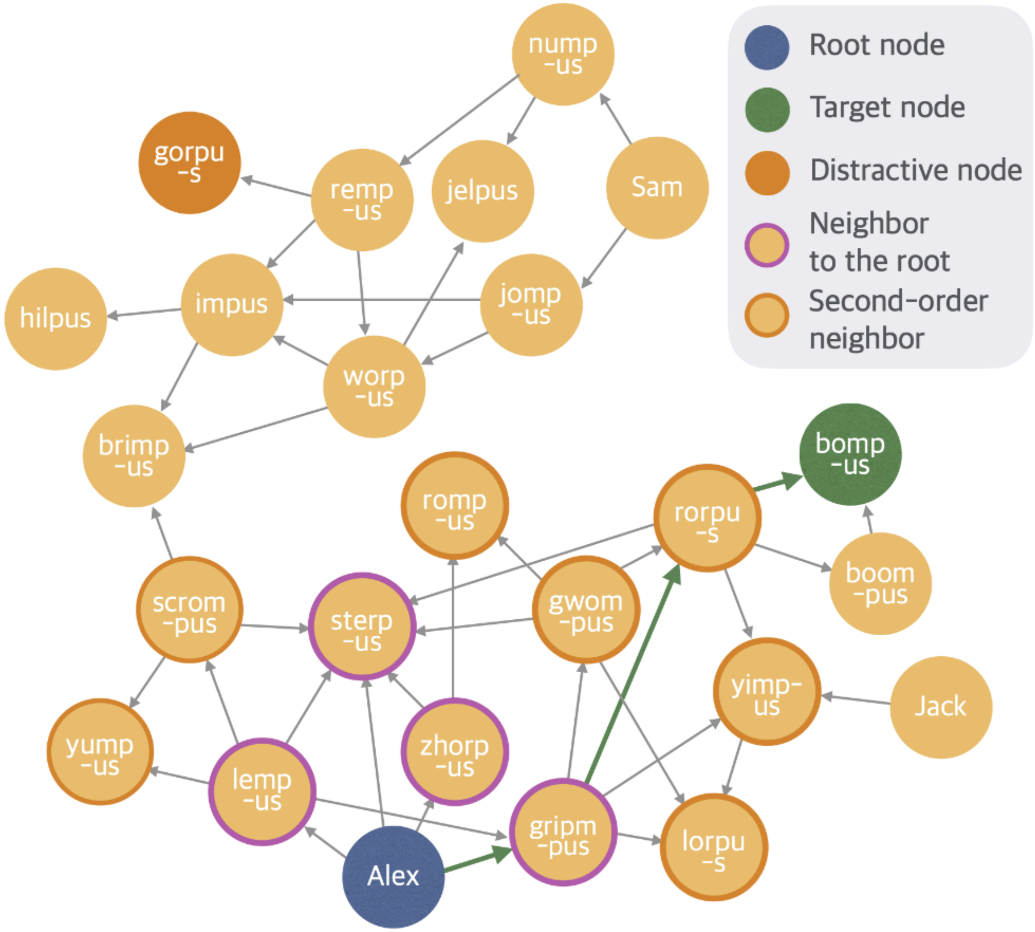
**Stage 1** [Question] <bot> [Thought] <eot> [Step 2] [Step 3] ··· [Step N] [Answer]

**Stage 2** [Question] <bot> [Thought] [Thought] <eot> [Step 3] ··· [Step N] [Answer]

··· ···

**Stage N** [Question] <bot> [Thought] [Thought] ··· [Thought] <eot> [Answer]

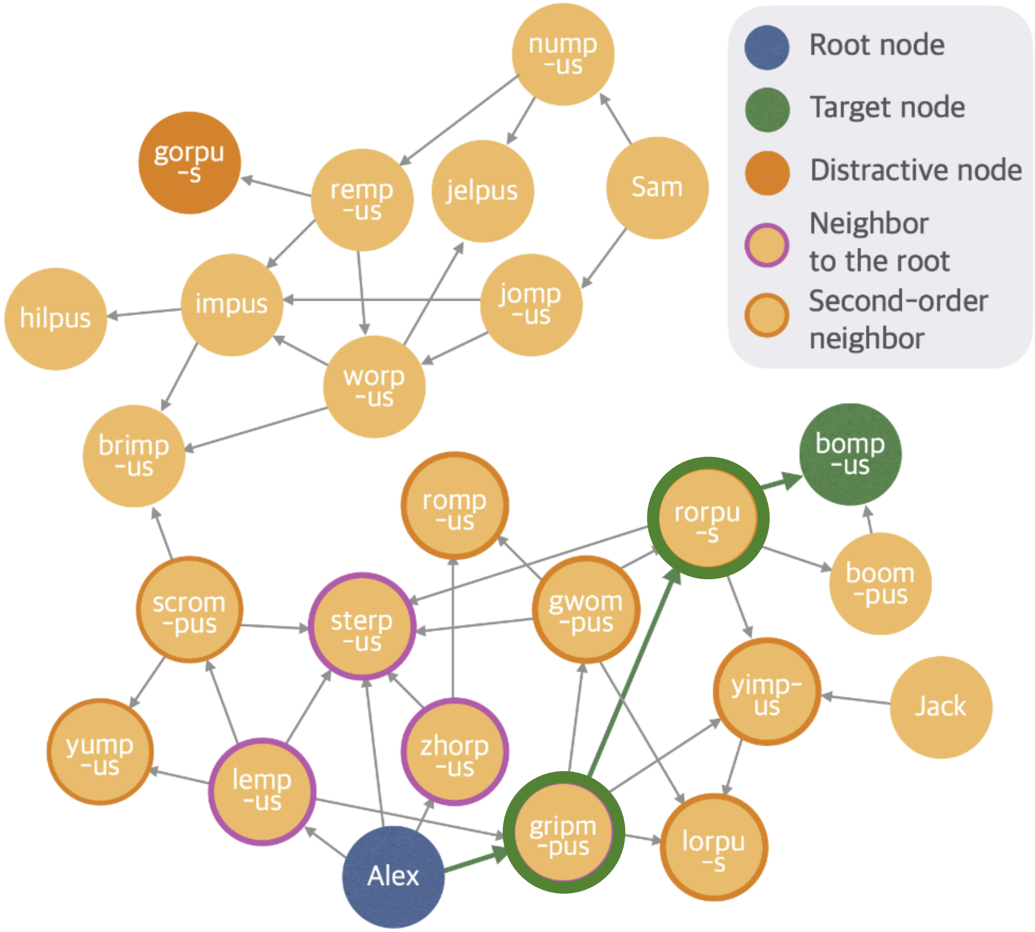
# Interpreting the embeddings




**Question**:

Every jells is a worpus. Sam is a jumpus. Every gwompus is a rompus. ⋯ Every lumps is a yumpus. Question: **Is Alex a gorpus or bompus?**

# Ground Truth Solutions



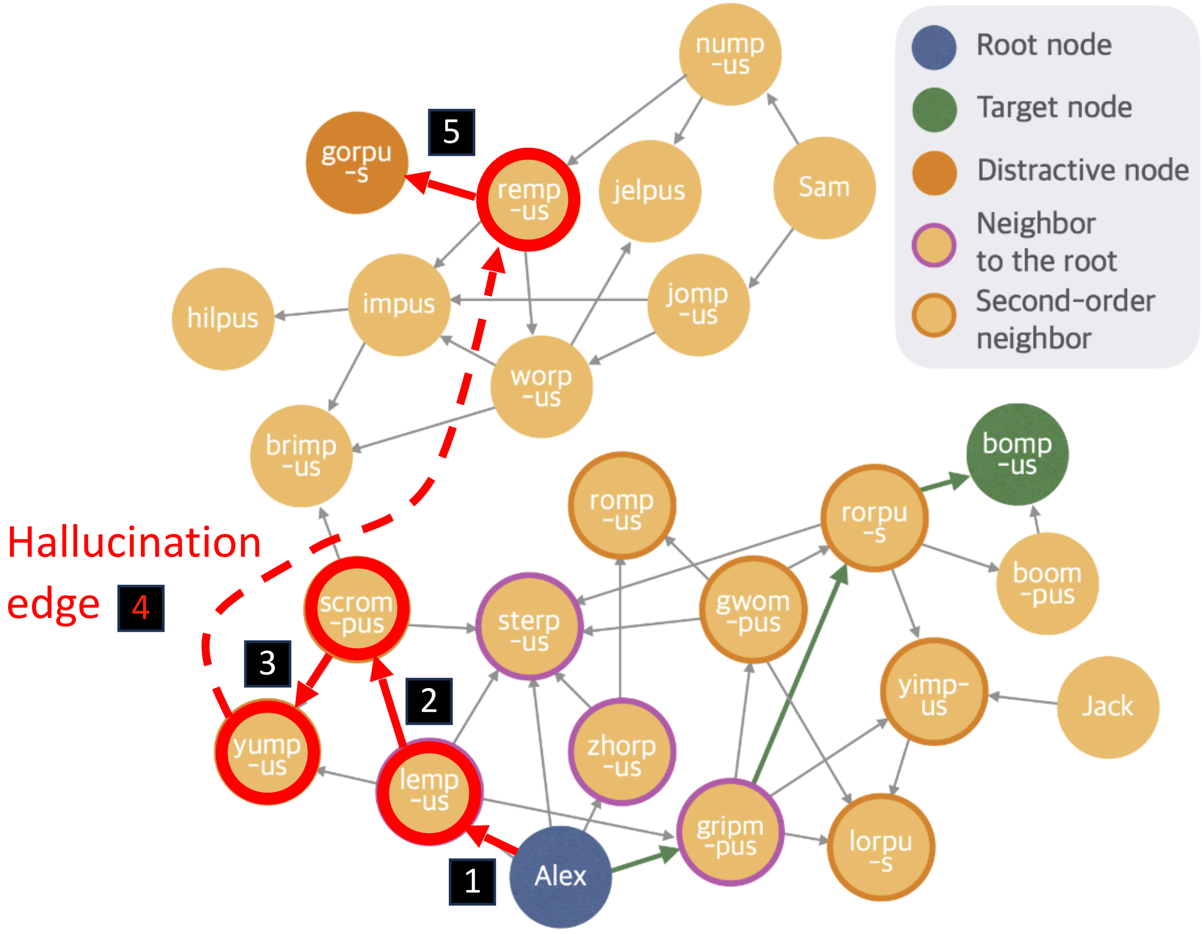

**Question:**

Every jells is a worpus. Sam is a jumpus. Every gwompus is a rompus. ⋯ Every lumps is a yumpus. Question: **Is Alex a gorpus or bompus?**

**Ground Truth Solution**

Alex is a grimpus.
Every grimpus is a rorpus.
Every rorpus is a bompus.
### Alex is a bompus

# Chain of thoughts lead to hallucinations




**Question**:

Every jells is a worpus. Sam is a jumpus. Every gwompus is a rompus. ⋯ Every lumps is a yumpus. Question: **Is Alex a gorpus or bompus?**

**Ground Truth Solution**

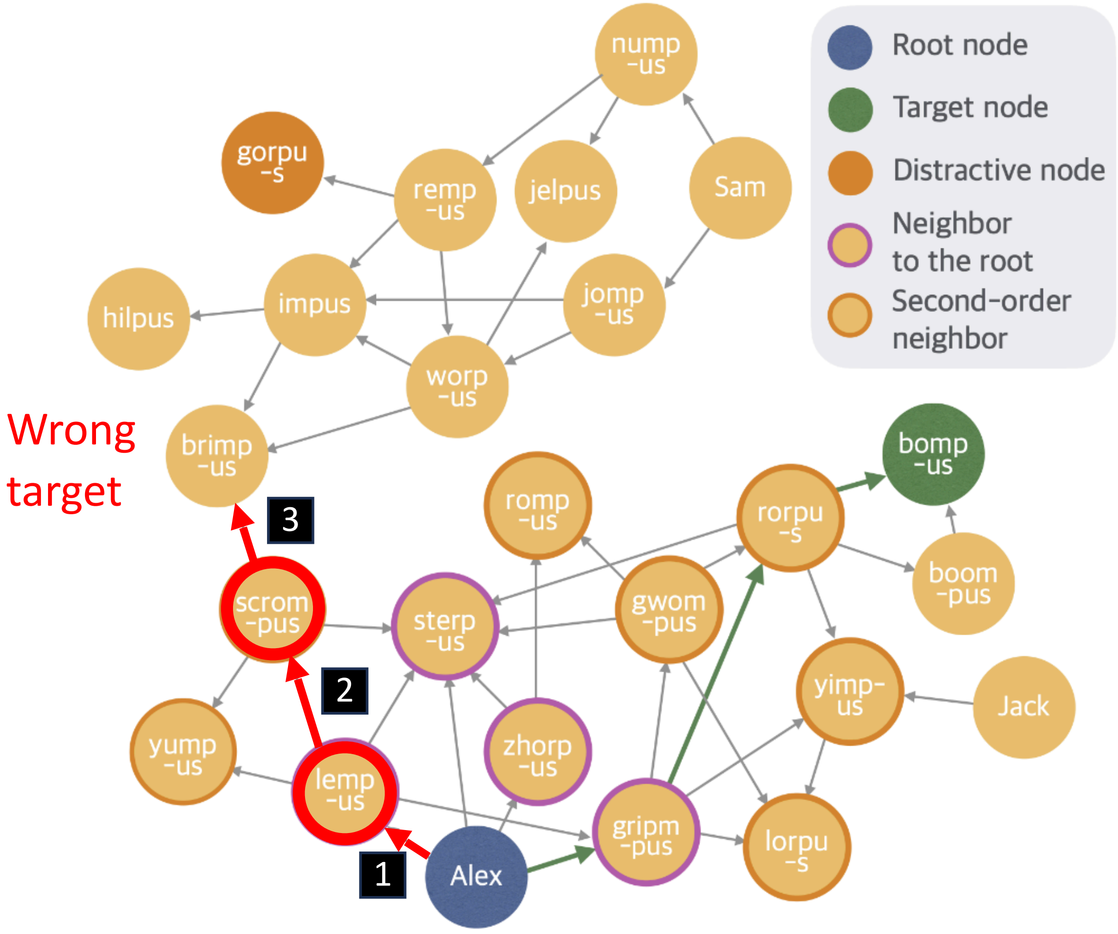
Alex is a grimpus.
Every grimpus is a rorpus.
Every rorpus is a bompus.
### Alex is a bompus

**CoT**

Alex is a lempus. 1
Every lempus is a scrompus. 2
Every scrompus is a yumpus. 3
Every yumpus is a rempus. 4
Every rempus is a gorpus. 5
### Alex is a gorpus ✗

*(Hallucination)*

# Continuous Thoughts




**Question**:

Every jells is a worpus. Sam is a jumpus. Every gwompus is a rompus. ⋯ Every lumps is a yumpus. Question: **Is Alex a gorpus or bompus?**

**Ground Truth Solution**

Alex is a grimpus.
Every grimpus is a rorpus.
Every rorpus is a bompus.
### Alex is a bompus

**CoT**

Alex is a lempus.
Every lempus is a scrompus.
Every scrompus is a yumpus.
Every yumpus is a rempus.
Every rempus is a gorpus.
### Alex is a gorpus ✗

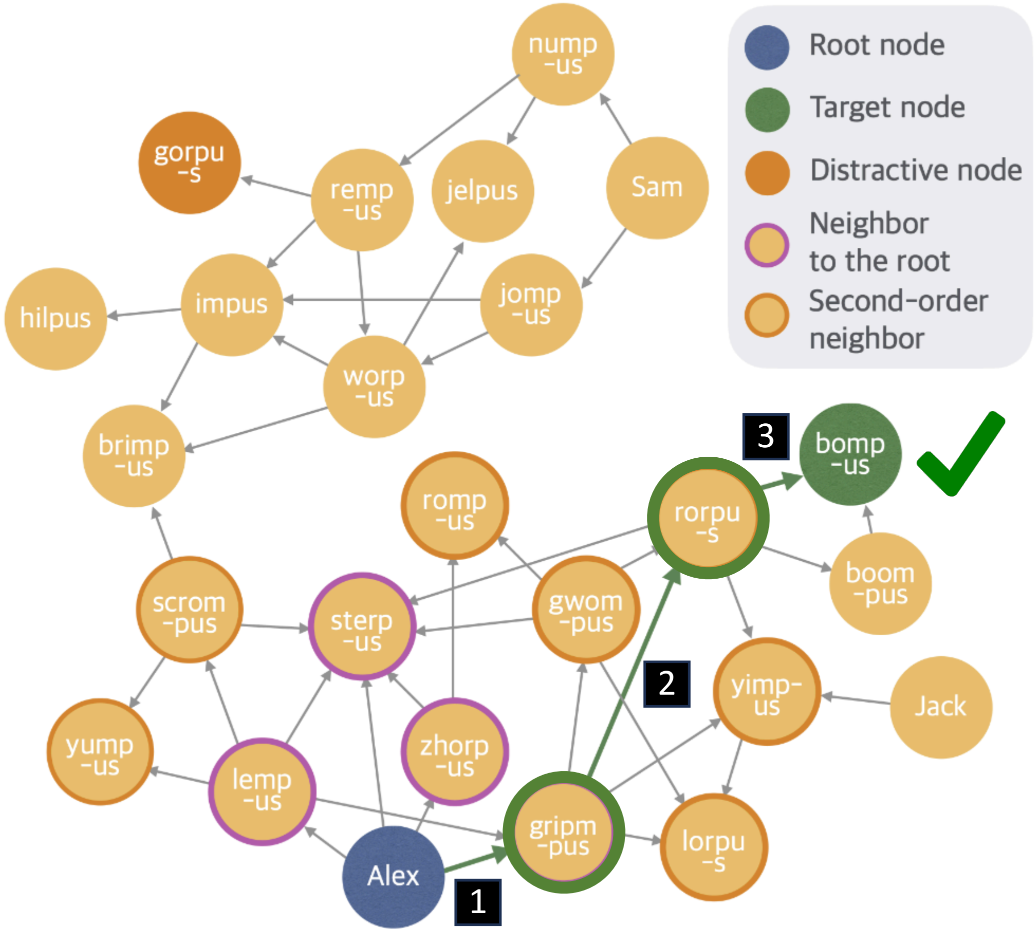
*(Hallucination)*

**Ours (k=1)**

<bot> [Thought] <eot> 1
Every lempus is a scrompus. 2
Every scrompus is a brimpus. 3
### Alex is a brimpus ✗

*(Wrong Target)*

# Two-step Continuous Thought works!




**Question**:

Every jells is a worpus. Sam is a jumpus. Every gwompus is a rompus. ⋯ Every lumps is a yumpus. Question: **Is Alex a gorpus or bompus?**

**Ground Truth Solution**

Alex is a grimpus.
Every grimpus is a rorpus.
Every rorpus is a bompus.
### Alex is a bompus

**CoT**

Alex is a lempus.
Every lempus is a scrompus.
Every scrompus is a yumpus.
Every yumpus is a rempus.
Every rempus is a gorpus.
### Alex is a gorpus ✗

(Hallucination)

**Ours (k=1)**

<bot> [Thought] <eot>
Every lempus is a scrompus.
Every scrompus is a brimpus.
### Alex is a brimpus ✗

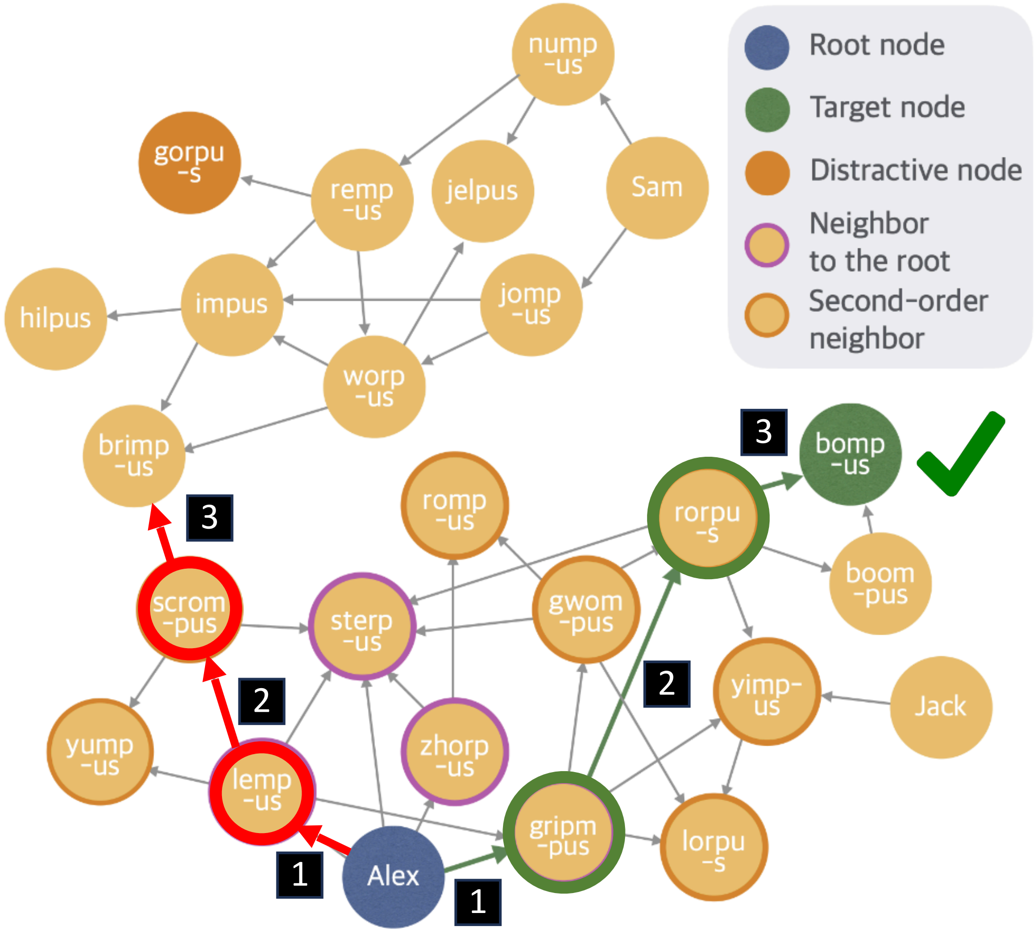
(Wrong Target)

**Ours (k=2)** 1 2

<bot> [thought] [thought] <eot>
Every rorpus is a bompus. 3
### Alex is a bompus ✓

*(Correct Path)*

# Two-step Continuous Thought works!




**Ours (k=1)** 1

<bot> [Thought] <eot>
Every lempus is a scrompus.
Every scrompus is a brimpus.
### Alex is a brimpus

(Wrong Target)

**Ours (k=2)** 1 2

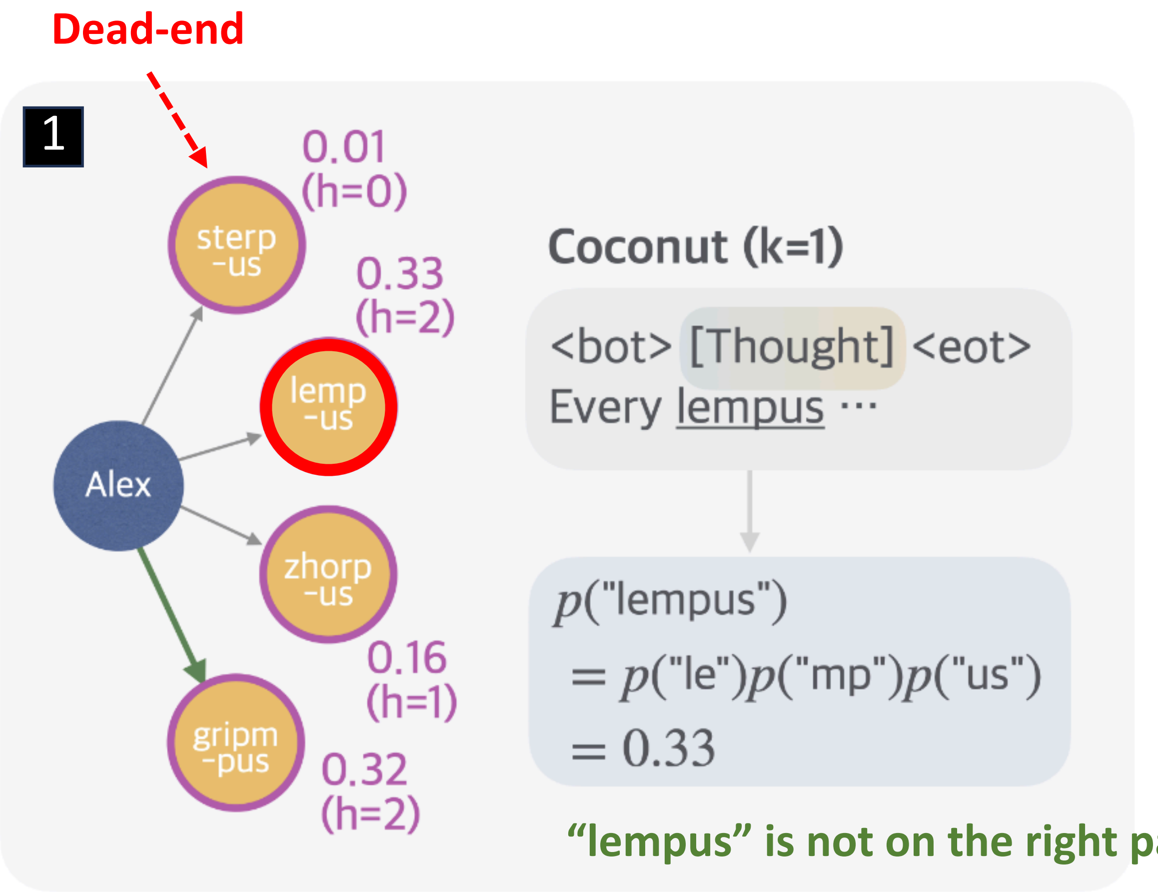
<bot> [thought] [thought] <eot>
Every rorpus is a bompus.
### Alex is a bompus

(Correct Path)

Why the same continuous thoughts 1 lead to different path?!

# What’s inside?

Let’s probe!

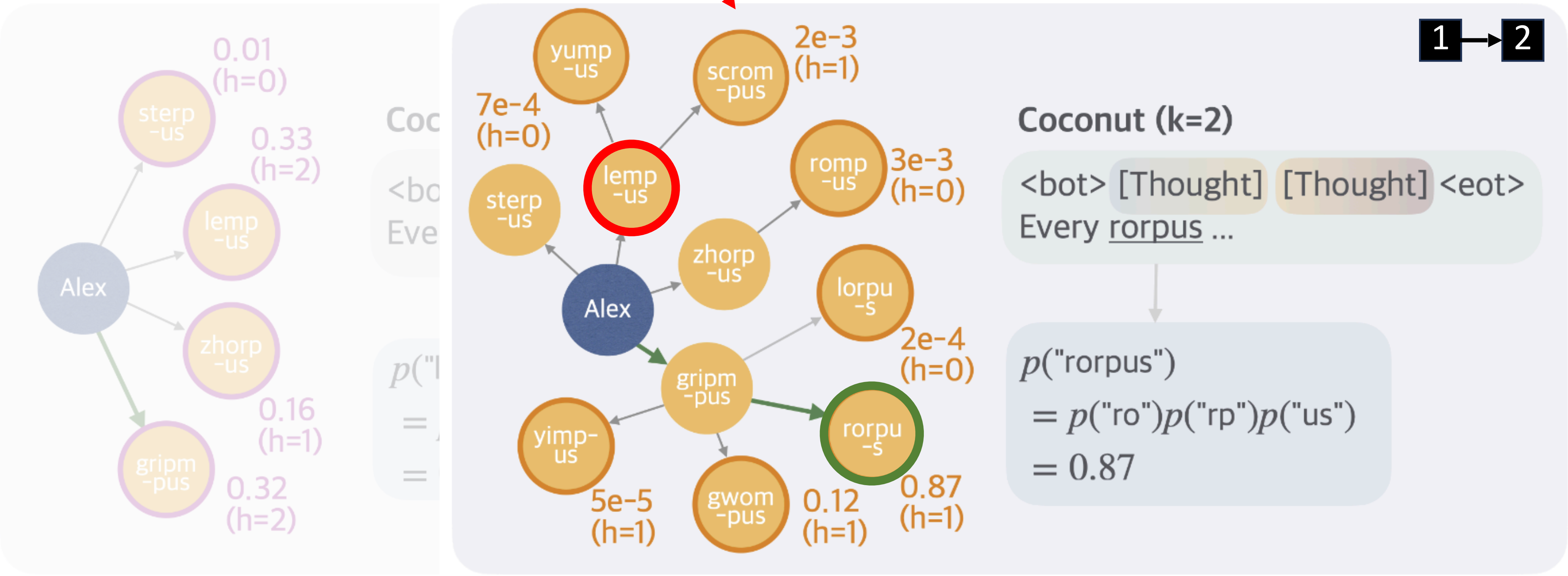



“lempus” is not on the right path but for step=1, it is the **most promising**

# What’s inside?

**Promising node → dead-end**

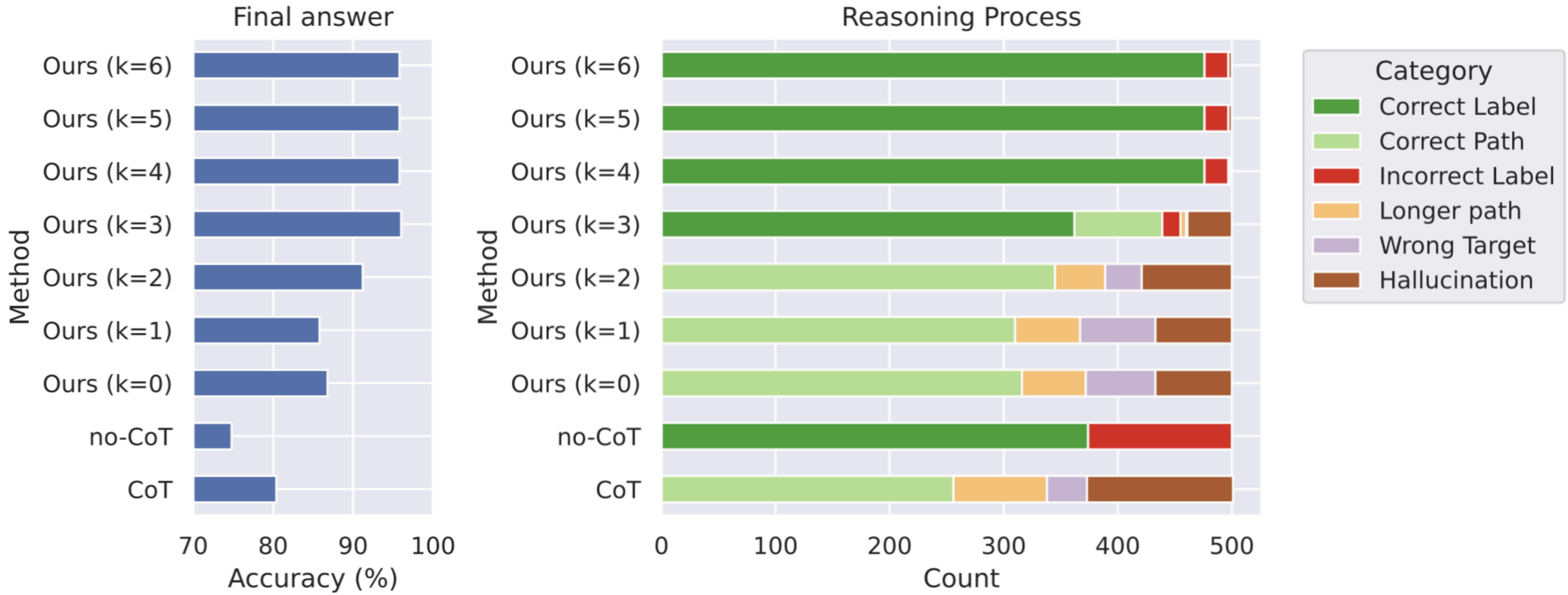
***Interestingly, it encodes all possible paths!***



# Performance in ProsQA

| Dataset | Training | Validation | Test |
|---|---|---|---|
| GSM8k | 385,620 | 500 | 1319 |
| ProntoQA | 9,000 | 200 | 800 |
| ProsQA | 17,886 | 300 | 500 |

| # Nodes | # Edges | Len. of Shortest Path | # Shortest Paths |
|---|---|---|---|
| 23.0 | 36.0 | 3.8 | 1.6 |



# CoConut

| Method | GSM8k | | ProntoQA | | ProsQA | |
|---|---|---|---|---|---|---|
| | Acc. (%) | # Tokens | Acc. (%) | # Tokens | Acc. (%) | # Tokens |
| CoT | 42.9 ±0.2 | 25.0 | 98.8 ±0.8 | 92.5 | 77.5 ±1.9 | 49.4 |
| No-CoT | 16.5 ±0.5 | 2.2 | 93.8 ±0.7 | 3.0 | 76.7 ±1.0 | 8.2 |
| iCoT | 30.0* | 2.2 | 99.8 ±0.3 | 3.0 | 98.2 ±0.3 | 8.2 |
| Pause Token | 16.4 ±1.8 | 2.2 | 77.7 ±21.0 | 3.0 | 75.9 ±0.7 | 8.2 |
| COCONUT (Ours) | 34.1 ±1.5 | 8.2 | 99.8 ±0.2 | 9.0 | 97.0 ±0.3 | 14.2 |
| - *w/o curriculum* | 14.4 ±0.8 | 8.2 | 52.4 ±0.4 | 9.0 | 76.1 ±0.2 | 14.2 |
| - *w/o thought* | 21.6 ±0.5 | 2.3 | 99.9 ±0.1 | 3.0 | 95.5 ±1.1 | 8.2 |
| - *pause as thought* | 24.1 ±0.7 | 2.2 | 100.0 ±0.1 | 3.0 | 96.6 ±0.8 | 8.2 |

Better performance than No-CoT
Shorter thinking process than CoT

# CoConut

| Method | GSM8k | | ProntoQA | | ProsQA | |
|---|---|---|---|---|---|---|
| | Acc. (%) | # Tokens | Acc. (%) | # Tokens | Acc. (%) | # Tokens |
| CoT | 42.9 ±0.2 | 25.0 | 98.8 ±0.8 | 92.5 | 77.5 ±1.9 | 49.4 |
| No-CoT | 16.5 ±0.5 | 2.2 | 93.8 ±0.7 | 3.0 | 76.7 ±1.0 | 8.2 |
| iCoT | 30.0* | 2.2 | 99.8 ±0.3 | 3.0 | 98.2 ±0.3 | 8.2 |
| Pause Token | 16.4 ±1.8 | 2.2 | 77.7 ±21.0 | 3.0 | 75.9 ±0.7 | 8.2 |
| COCONUT (Ours) | $\boxed{34.1 \pm 1.5}$ | 8.2 | 99.8 ±0.2 | 9.0 | 97.0 ±0.3 | 14.2 |
| - *w/o curriculum* | 14.4 ±0.8 | 8.2 | 52.4 ±0.4 | 9.0 | 76.1 ±0.2 | 14.2 |
| - *w/o thought* | 21.6 ±0.5 | 2.3 | 99.9 ±0.1 | 3.0 | 95.5 ±1.1 | 8.2 |
| - *pause as thought* | 24.1 ±0.7 | 2.2 | [illegible] | [illegible] | [illegible] | [illegible] |

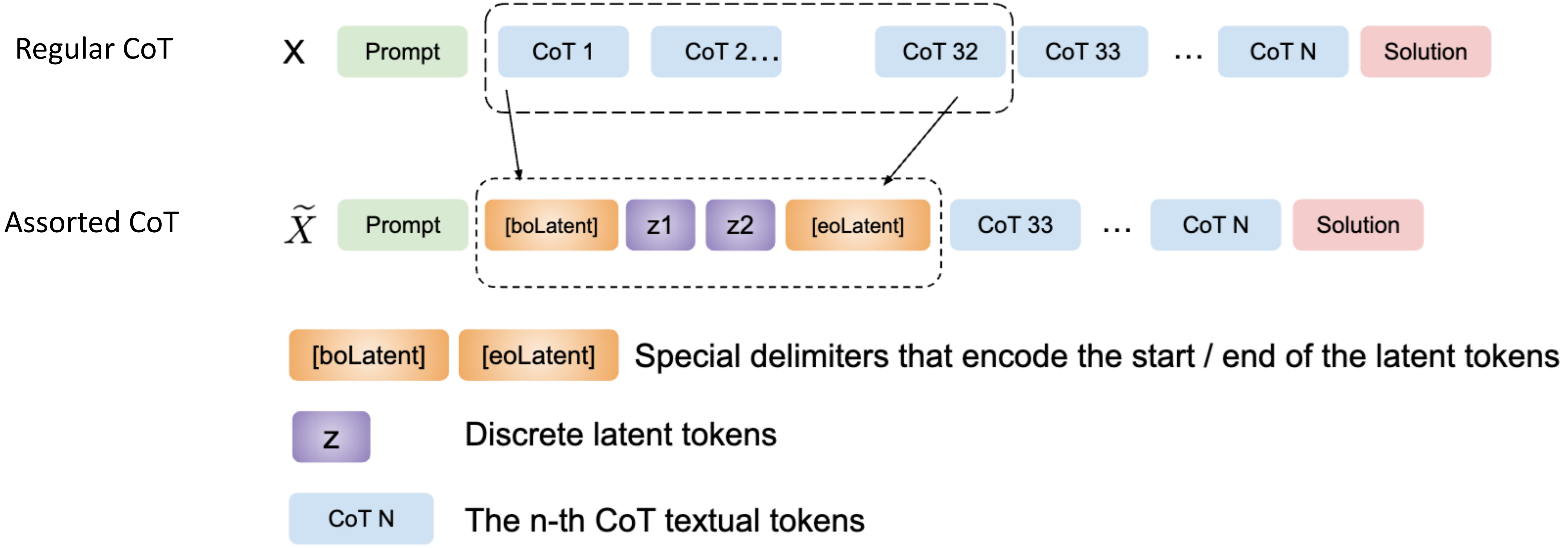
Better performance than No-CoT
Shorter thinking process than CoT

**Cons**

**1. Latent tokens are not interpretable**

**2. Only tested on GSM8k**

# Token Assorted



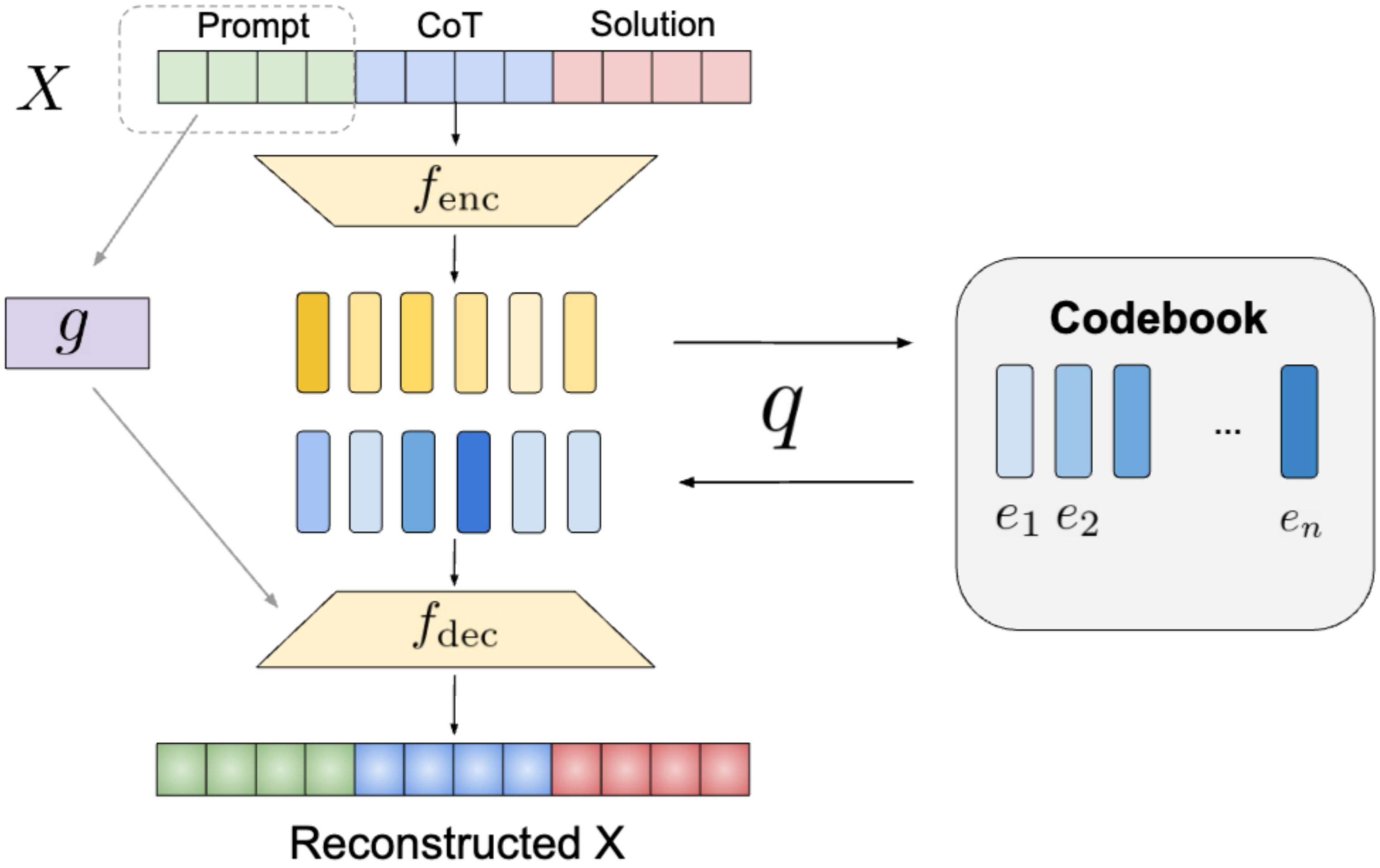



[D. Su et al, *Token Assorted: Mixing Latent and Text Tokens for Improved Language Model Reasoning*, ICML'25]

# Token Assorted

How the latent codes are constructed?

Using VQVAE



# Better Performance

| Model | | In-Domain | | Out-of-Domain | | | | | Average |
|---|---|---|---|---|---|---|---|---|---|
| | | Math | GSM8K | Gaokao-Math-2023 | DM-Math | College-Math | Olympia-Math | TheoremQA | All Datasets |
| **Llama-3.2-1B** | Sol-Only | 4.7 | 6.8 | 0.0 | 10.4 | 5.3 | 1.3 | 3.9 | 4.6 |
| | CoT | 10.5 | 42.7 | **10.0** | 3.4 | 17.1 | 1.5 | 9.8 | 14.1 |
| | iCoT | 8.2 | 10.5 | 3.3 | 11.3 | 7.6 | **2.1** | 10.7 | 7.7 |
| | Pause Token | 5.1 | 5.3 | 2.0 | 1.4 | 0.5 | 0.0 | 0.6 | 2.1 |
| | **Latent (ours)** | **14.7** (↑ +4.2) | **48.7** (↑ +6) | **10.0** | **14.6** (↑ +3.3) | **20.5** (↑ +3.4) | 1.8 | **11.3** (↑ +0.6) | **17.8** (↑ +3.7) |
| **Llama-3.2-3B** | Sol-Only | 6.1 | 8.1 | 3.3 | 14.0 | 7.0 | 1.8 | 6.8 | 6.7 |
| | CoT | 21.9 | 69.7 | 16.7 | **27.3** | 30.9 | 2.2 | 11.6 | 25.2 |
| | iCoT | 12.6 | 17.3 | 3.3 | 16.0 | 14.2 | **4.9** | **13.9** | 11.7 |
| | Pause Token | 25.2 | 53.7 | 4.1 | 7.4 | 11.8 | 0.7 | 1.0 | 14.8 |
| | **Latent (ours)** | **26.1** (↑ +4.2) | **73.8** (↑ +4.1) | **23.3** (↑ +6.6) | 27.1 | **32.9** (↑ +2) | 4.2 | 13.5 | **28.1** (↑ +2.9) |
| **Llama-3.1-8B** | Sol-Only | 11.5 | 11.8 | 3.3 | 17.4 | 13.0 | 3.8 | 6.7 | 9.6 |
| | CoT | 32.9 | 80.1 | 16.7 | 39.3 | 41.9 | 7.3 | 15.8 | 33.4 |
| | iCoT | 17.8 | 29.6 | 16.7 | 20.3 | 21.3 | 7.6 | 14.8 | 18.3 |
| | Pause Token | **39.6** | 79.5 | 6.1 | 25.4 | 25.1 | 1.3 | 4.0 | 25.9 |
| | **Latent (ours)** | 37.2 | **84.1** (↑ +4.0) | **30.0** (↑ +13.3) | **41.3** (↑ +2) | **44.0** (↑ +2.1) | **10.2** (↑ +2.6) | **18.4** (↑ +2.6) | **37.9** (↑ +4.5) |

# Shorter CoT

| Model | | In-Domain (# of tokens) | | Out-of-Domain (# of tokens) | | | | | Average |
|---|---|---|---|---|---|---|---|---|---|
| | | Math | GSM8K | Gaokao-Math-2023 | DM-Math | College-Math | Olympia-Math | TheoremQA | All Datasets |
| **Llama-3.2-1B** | Sol-Only | 4.7 | 6.8 | 0.0 | 10.4 | 5.3 | 1.3 | 3.9 | 4.6 |
| | CoT | 646.1 | 190.3 | 842.3 | 578.7 | 505.6 | 1087.0 | 736.5 | 655.2 |
| | iCoT | 328.4 | 39.8 | 354.0 | 170.8 | 278.7 | 839.4 | 575.4 | 369.5 |
| | Pause Token | 638.8 | 176.4 | 416.1 | 579.9 | 193.8 | 471.9 | 988.1 | 495 |
| | **Latent (ours)** | 501.6 **(↓ -22%)** | 181.3 **(↓ -5%)** | 760.5 **(↓ -11%)** | 380.1 **(↓ -34%)** | 387.3 **(↓ -23%)** | 840.0 **(↓ -22%)** | 575.5 **(↓ -22%)** | 518 **(↓ -21%)** |
| **Llama-3.2-3B** | Sol-Only | 6.1 | 8.1 | 3.3 | 14.0 | 7.0 | 1.8 | 6.8 | 6.7 |
| | CoT | 649.9 | 212.1 | 823.3 | 392.8 | 495.9 | 1166.7 | 759.6 | 642.9 |
| | iCoT | 344.4 | 60.7 | 564.0 | 154.3 | 224.9 | 697.6 | 363.6 | 344.2 |
| | Pause Token | 307.9 | 162.3 | 108.9 | 251.5 | 500.96 | 959.5 | 212.8 | 354.7 |
| | **Latent (ours)** | 516.7 **(↓ -20%)** | 198.8 **(↓ -6%)** | 618.5 **(↓ -25%)** | 340.0 **(↓ -13%)** | 418.0 **(↓ -16%)** | 832.8 **(↓ -29%)** | 670.2 **(↓ -12%)** | 513.6 **(↓ -20%)** |
| **Llama-3.1-8B** | Sol-Only | 11.5 | 11.8 | 3.3 | 17.4 | 13.0 | 3.8 | 6.7 | 9.6 |
| | CoT | 624.3 | 209.5 | 555.9 | 321.8 | 474.3 | 1103.3 | 760.1 | 578.5 |
| | iCoT | 403.5 | 67.3 | 444.8 | 137.0 | 257.1 | 797.1 | 430.9 | 362.5 |
| | Pause Token | 469.4 | 119.0 | 752.6 | 413.4 | 357.3 | 648.2 | 600.1 | 480 |
| | **Latent (ours)** | 571.9 **(↓ -9 %)** | 193.9 **(↓ -8 %)** | 545.8 **(↓ -2 %)** | 292.1 **(↓ -10%)** | 440.3 **(↓ -8%)** | 913.7 **(↓ -17 %)** | 637.2 **(↓ -16 %)** | 513.7 **(↓ -10%)** |

# Main Theorem

**Theorem (informal)**

For $n$-vertex directed graphs, a **2-layer** transformer with continuous CoT can solve reachability using $O(n)$ decoding steps with $O(n)$ embedding dimensions.

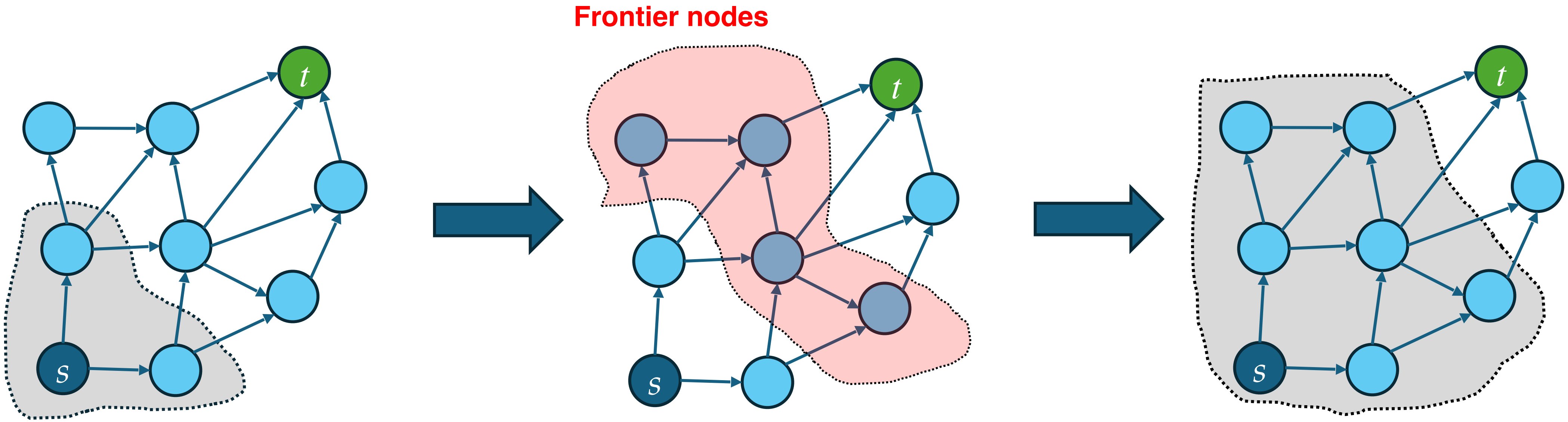
**Best known results for discrete CoT:** $O(n^2)$

**Secret Sauce:** Superposition of the embeddings!



[H. Zhu et al, *Reasoning by Superposition: A Theoretical Perspective on Chain of Continuous Thought*, NeurIPS'25]

# Continuous CoT: Decoding as search




$$[t_1] = \frac{1}{\sqrt{|\mathcal{V}_1(s)|}} \sum_{v \in \mathcal{V}_1} \vec{u}_v$$

The embedding contains **superposition**

$$[t_2] = \frac{1}{\sqrt{|\mathcal{V}_2(s)|}} \sum_{v \in \mathcal{V}_2} \vec{u}_v$$



[H. Zhu et al, *Reasoning by Superposition: A Theoretical Perspective on Chain of Continuous Thought*, NeurIPS'25]

# Mechanism of Emerged Representation

Representation

(Traditional) Symbolic representation

### 4 Conclusions

We have extended the GLU family of layers and proposed their use in Transformer. In a transfer-learning setup, the new variants seem to produce better perplexities for the de-noising objective used in pre-training, as well as better results on many downstream language-understanding tasks. These architectures are simple to implement, and have no apparent computational drawbacks. We offer no explanation as to why these architectures seem to work; we attribute their success, as all else, to divine benevolence.

Neural Representation🤔

*Shall we just acknowledge that as "**divine benevolence**"?*

# Mechanism of Emerged Representation

Representation

(Traditional) Symbolic representation

Neural Representation🤔

Emerging Symbolic Structure

# Modular Addition

$$a + b = c \bmod d$$

Does neural network have an *implicit table* to do retrieval?

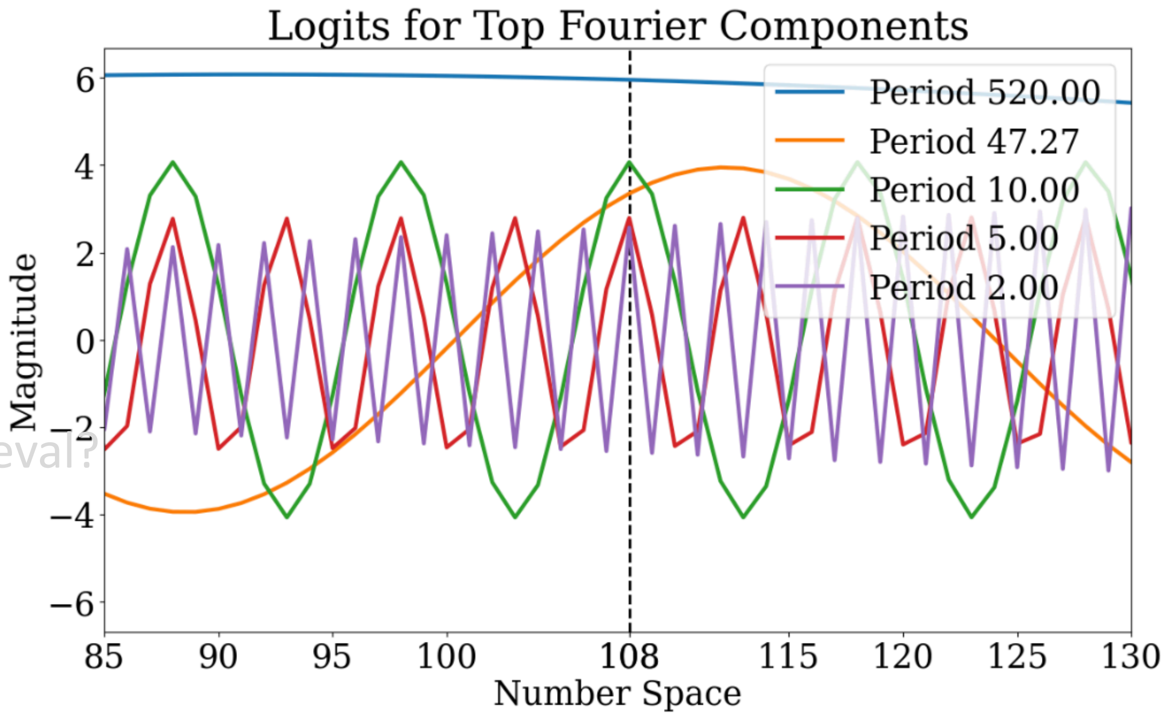
# Modular Addition

$$a + b = c \bmod d$$

Does neural network have an *implicit table* to do retrieval?

Learned representation = Fourier basis 🤯

**Why?** 🤔

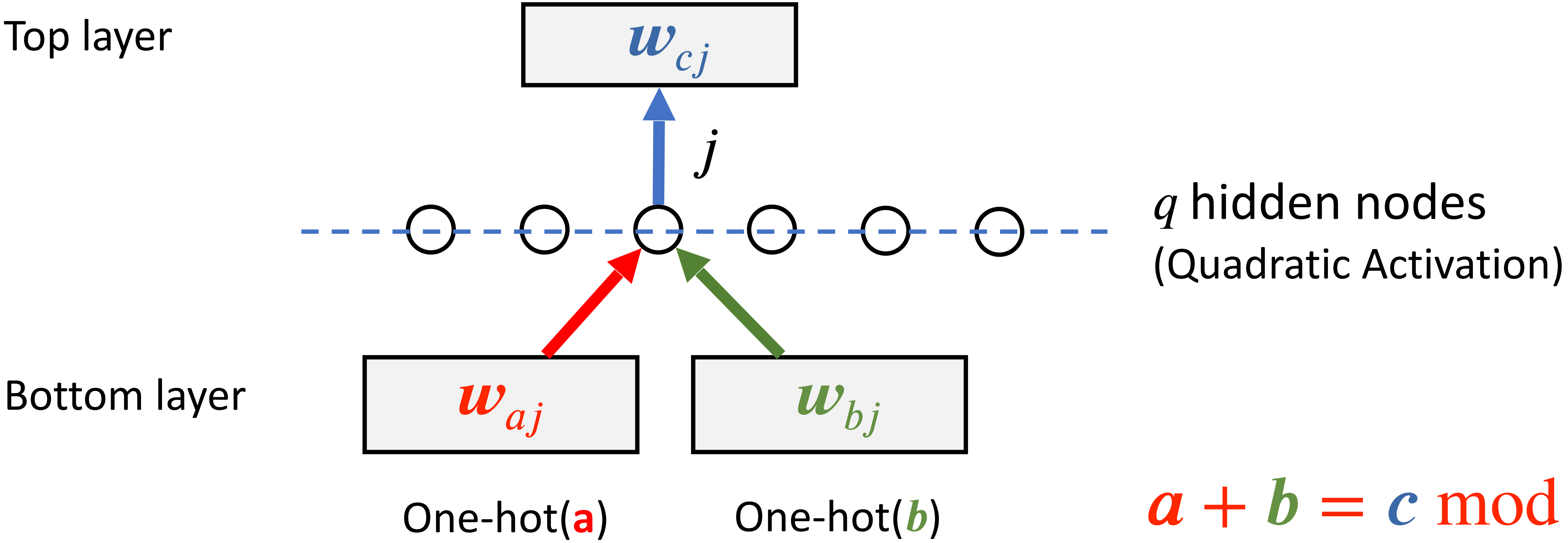



(a) Final logits for top Fourier components


[T. Zhou et al, *Pre-trained Large Language Models Use Fourier Features to Compute Addition*, NeurIPS'24]
[S. Kantamneni, *Language Models Use Trigonometry to Do Addition*, arXiv'25]

# Minimal Problem Setup

**MSE Loss:** $Min \left\| \text{Output} - \text{one-hot}(c) \right\|_2$



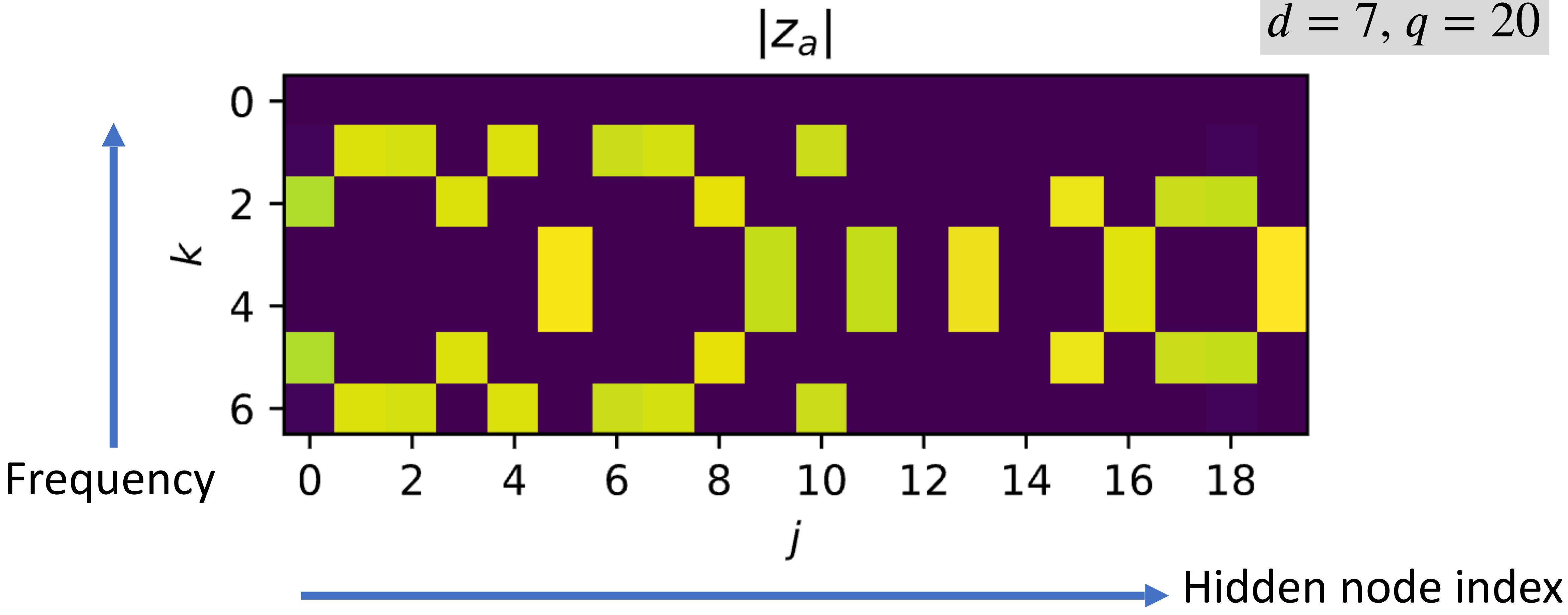

$a + b = c \bmod d$



[Y. Tian, *Composing Global Solutions to Reasoning Tasks via Algebraic Objects in Neural Nets*, NeurIPS'25]

# What a Gradient Descent Solution look like?

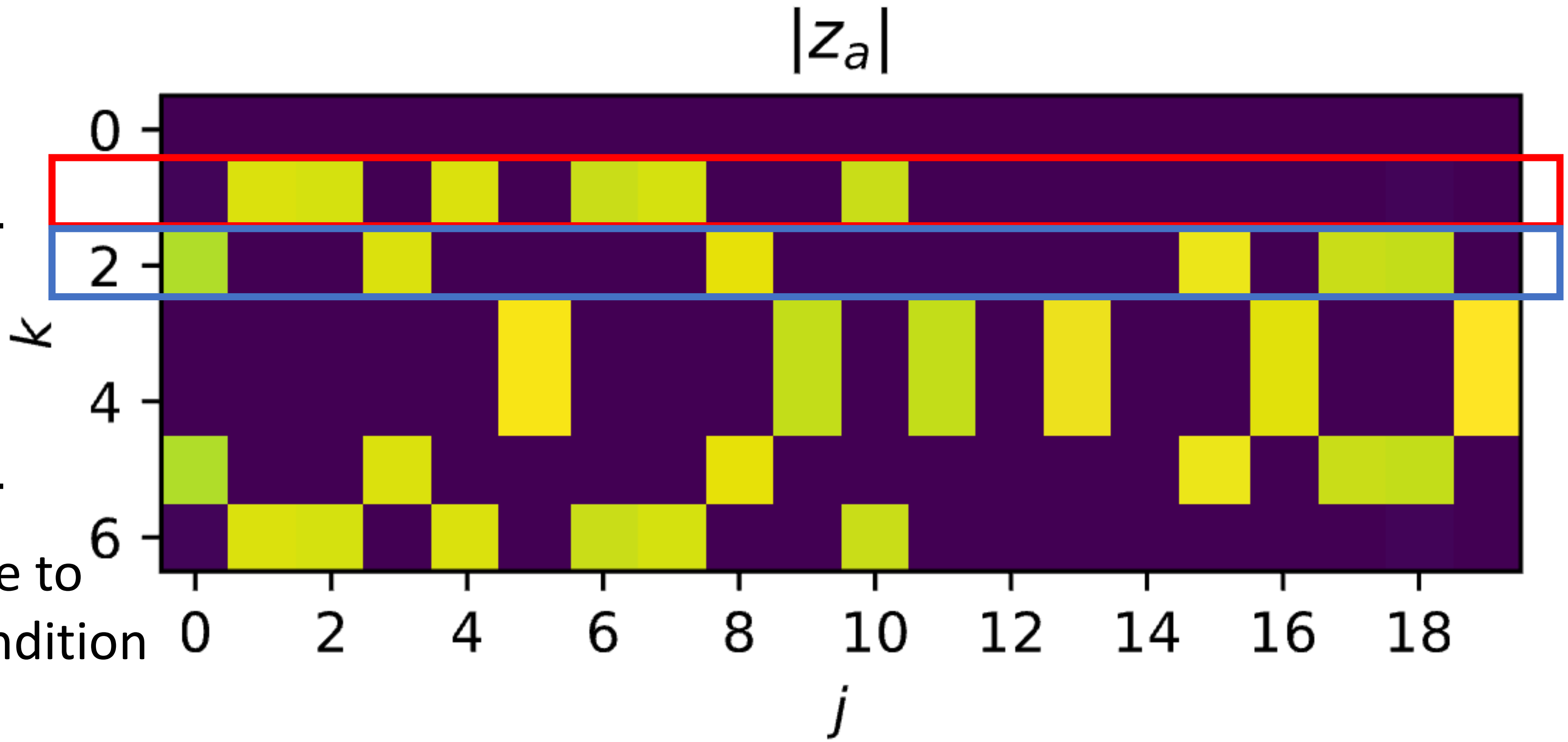
$d = 7, q = 20$






[Y. Tian, *Composing Global Solutions to Reasoning Tasks via Algebraic Objects in Neural Nets*, NeurIPS'25]

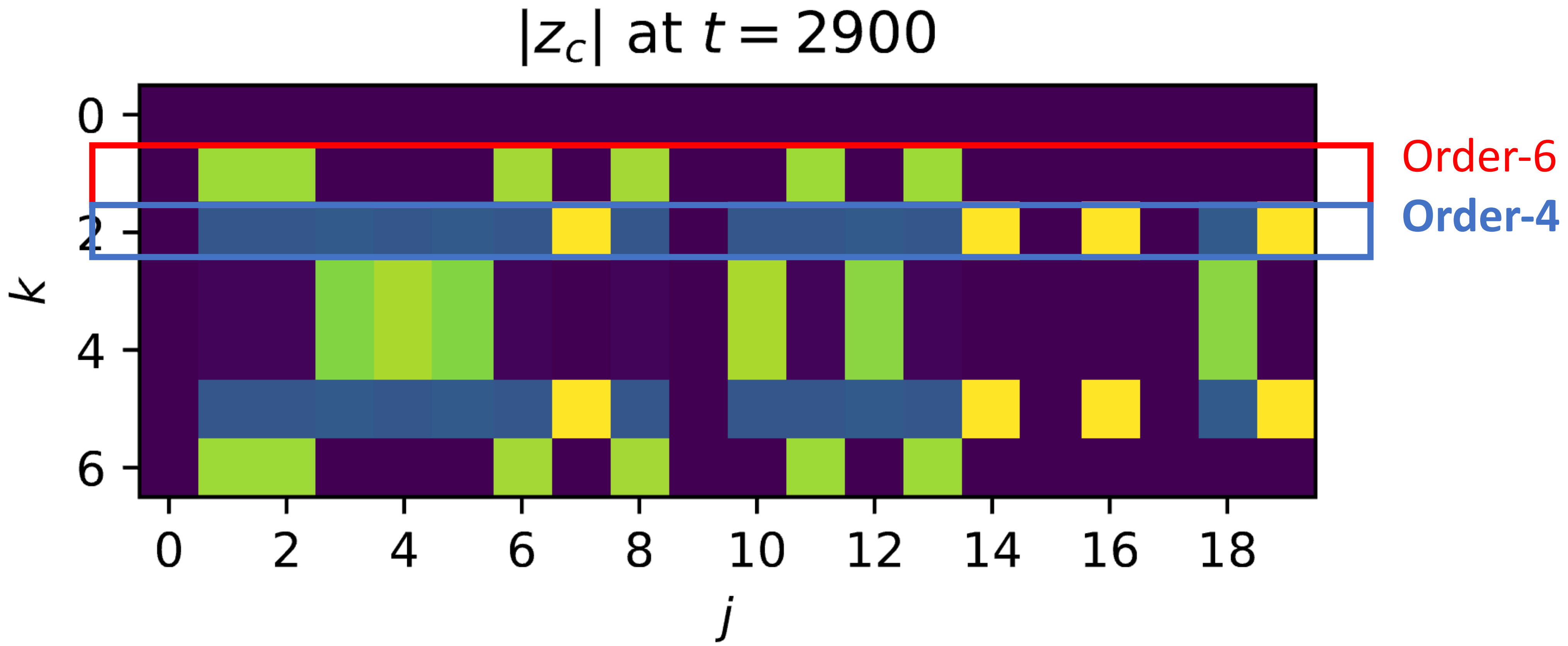
# What a Gradient Descent Solution look like?



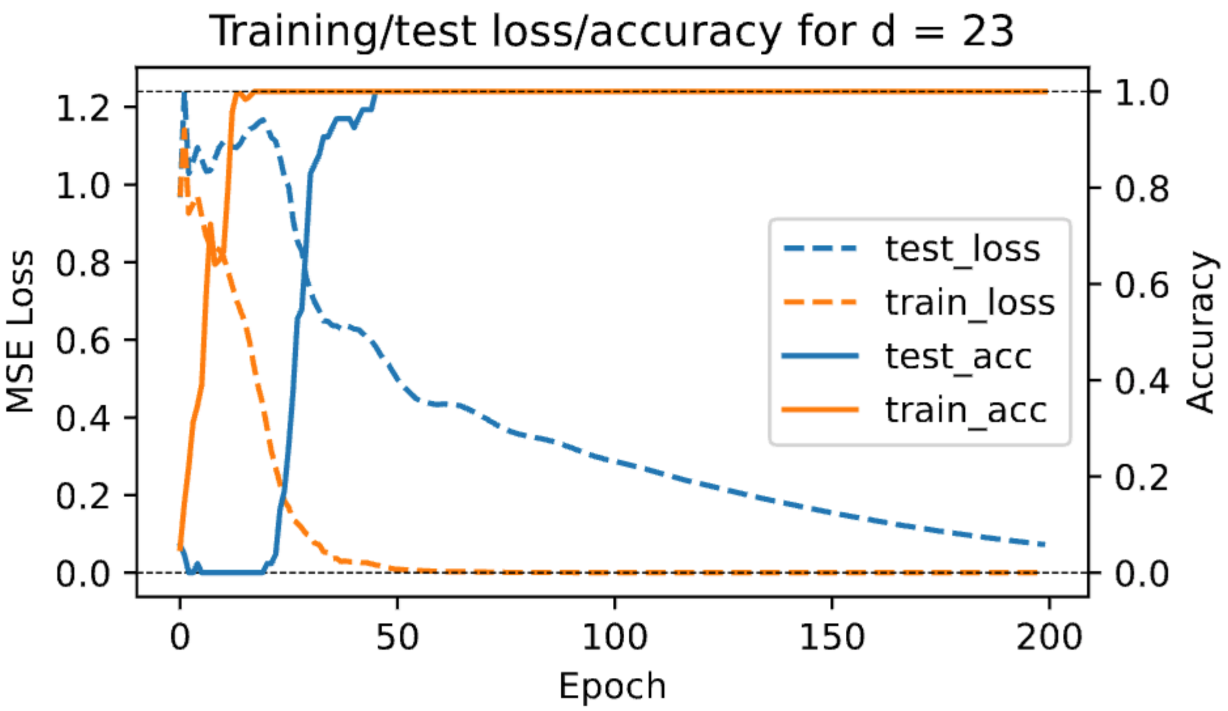

Order-6 solutions

Symmetry due to Hermitian condition

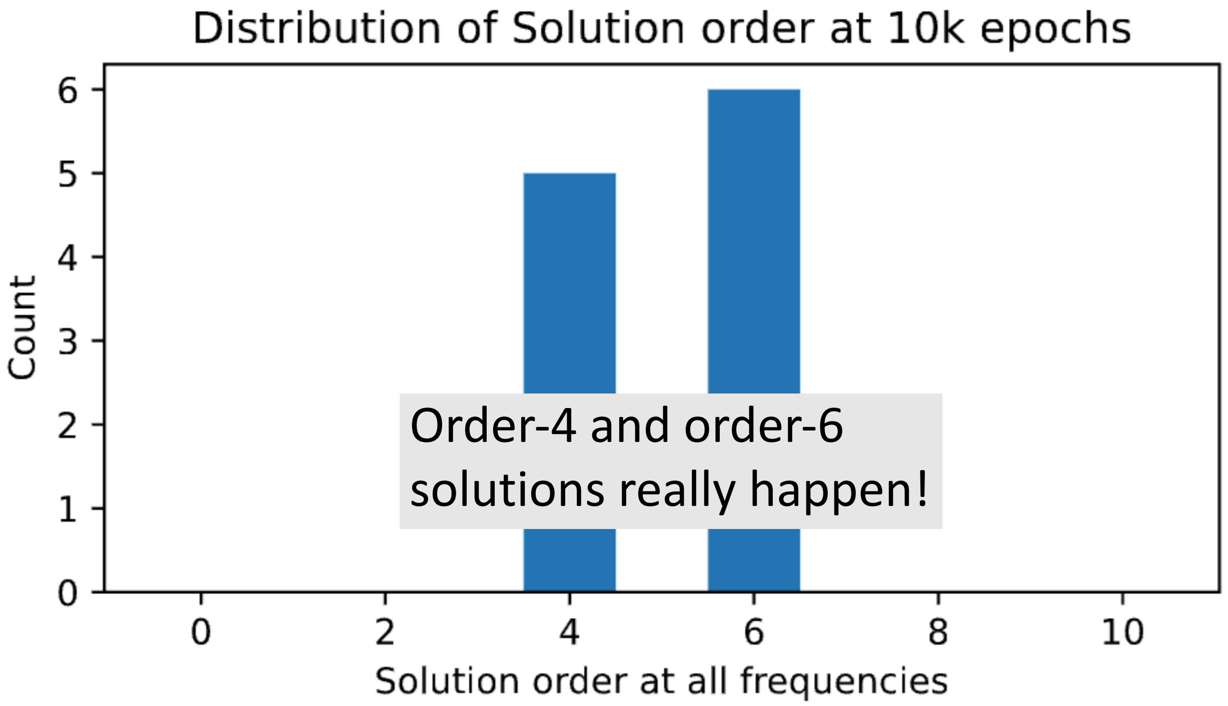
# What a Gradient Descent Solution look like?



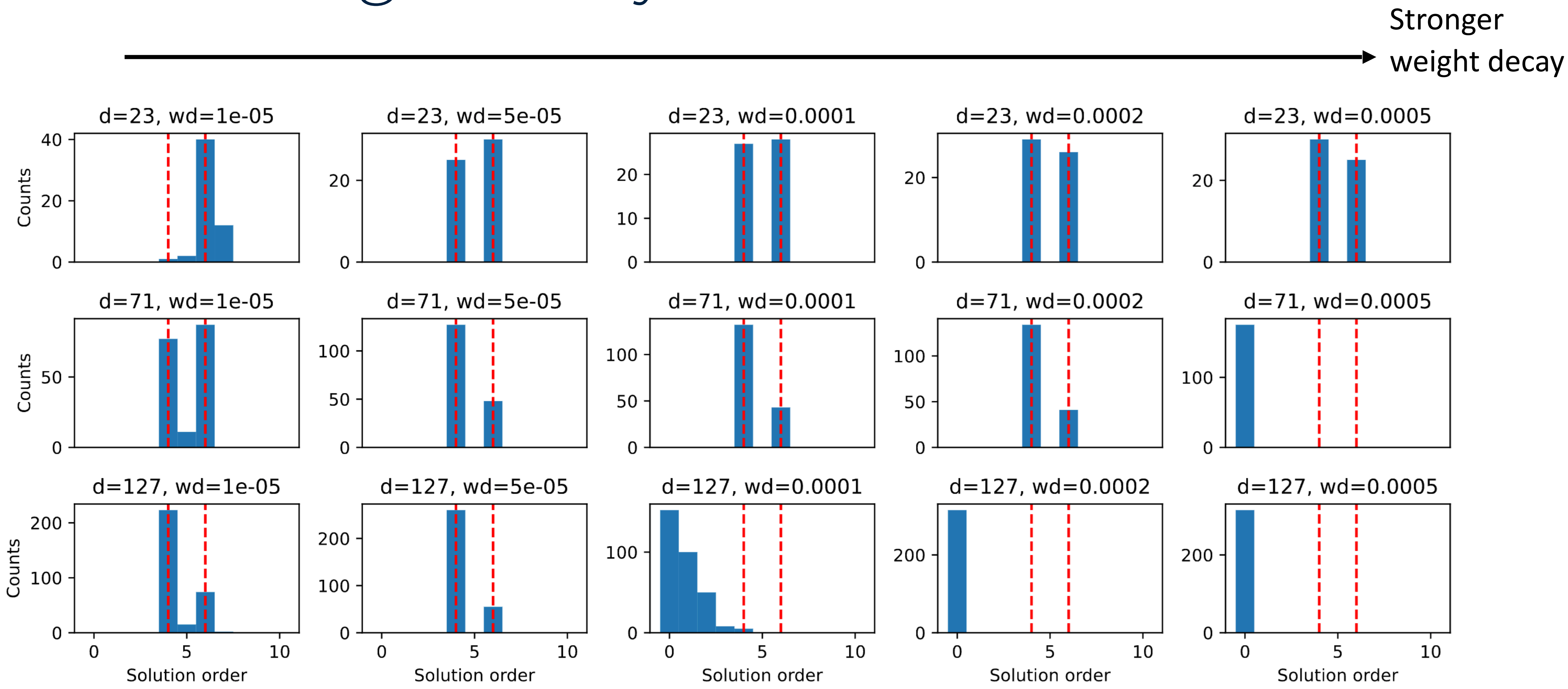
# More Statistics on Gradient Descent Solutions




Grokking Behaviors!



# Effect of Weight Decay



# Theory to explicitly construct such solutions

Order-6 $\boldsymbol{z}_{F6}$ (2*3)

$$\boldsymbol{z}_{F6} = \frac{1}{\sqrt[3]{6}} \sum_{k=1}^{(d-1)/2} \boldsymbol{z}_{\mathrm{syn}}^{(k)} * \boldsymbol{z}_{\nu}^{(k)} * \boldsymbol{y}_k$$

# Theory to explicitly construct such solutions

Order-6 $\boldsymbol{z}_{F6}$ (2*3)

$$\boldsymbol{z}_{F6} = \frac{1}{\sqrt[3]{6}} \sum_{k=1}^{(d-1)/2} \boldsymbol{z}_{\mathrm{syn}}^{(k)} * \boldsymbol{z}_{\nu}^{(k)} * \boldsymbol{y}_k$$

Order-4 $\boldsymbol{z}_{F4/6}$ (2*2)
(mixed with order-6)

$$\boldsymbol{z}_{F4/6} = \frac{1}{\sqrt[3]{6}} \hat{\boldsymbol{z}}_{F6}^{(k_0)} + \frac{1}{\sqrt[3]{4}} \sum_{k=1, k\neq k_0}^{(d-1)/2} \boldsymbol{z}_{F4}^{(k)}$$

# Theory to explicitly construct such solutions

Order-6 $\boldsymbol{z}_{F6}$ (2*3)

$$\boldsymbol{z}_{F6} = \frac{1}{\sqrt[3]{6}} \sum_{k=1}^{(d-1)/2} \boldsymbol{z}_{\mathrm{syn}}^{(k)} * \boldsymbol{z}_{\nu}^{(k)} * \boldsymbol{y}_k$$

Order-4 $\boldsymbol{z}_{F4/6}$ (2*2)
(mixed with order-6)

$$\boldsymbol{z}_{F4/6} = \frac{1}{\sqrt[3]{6}} \hat{\boldsymbol{z}}_{F6}^{(k_0)} + \frac{1}{\sqrt[3]{4}} \sum_{k=1, k \neq k_0}^{(d-1)/2} \boldsymbol{z}_{F4}^{(k)}$$

Perfect memorization
(order-d per frequency)

$$\boldsymbol{z}_a = \sum_{j=0}^{d-1} \boldsymbol{u}_a^j, \qquad \boldsymbol{z}_b = \sum_{j=0}^{d-1} \boldsymbol{u}_b^j$$

$$\boldsymbol{z}_M = d^{-2/3} \boldsymbol{z}_a * \boldsymbol{z}_b$$

# Gradient Descent solutions matches with construction

| $d$ | %not | %non-factorable | | error ($\times 10^{-2}$) | | solution distribution (%) in factorable ones | | | |
|---|---|---|---|---|---|---|---|---|---|
| | order-4/6 | order-4 | order-6 | order-4 | order-6 | $\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\xi}$ | $\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\mathrm{syn},\alpha\beta}$ | $\boldsymbol{z}^{(k)}_{\nu} * \boldsymbol{z}^{(k)}_{\mathrm{syn}}$ | others |
| 23 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $5.71_{\pm 5.71}$ | $0.05_{\pm 0.01}$ | $4.80_{\pm 0.96}$ | $47.07_{\pm 1.88}$ | $11.31_{\pm 1.76}$ | $39.80_{\pm 2.11}$ | $1.82_{\pm 1.82}$ |
| 71 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $0.00_{\pm 0.00}$ | $0.03_{\pm 0.00}$ | $5.02_{\pm 0.25}$ | $72.57_{\pm 0.70}$ | $4.00_{\pm 1.14}$ | $21.14_{\pm 2.14}$ | $2.29_{\pm 1.07}$ |
| 127 | $0.0_{\pm 0.0}$ | $1.50_{\pm 0.92}$ | $0.00_{\pm 0.00}$ | $0.26_{\pm 0.14}$ | $0.93_{\pm 0.18}$ | $82.96_{\pm 0.39}$ | $2.25_{\pm 0.64}$ | $14.13_{\pm 0.87}$ | $0.66_{\pm 0.66}$ |

$q = 512,\ wd = 5 \cdot 10^{-5}$

# Gradient Descent solutions matches with construction

| $d$ | %not | %non-factorable | | error ($\times 10^{-2}$) | | solution distribution (%) in factorable ones | | | |
|---|---|---|---|---|---|---|---|---|---|
| | order-4/6 | order-4 | order-6 | order-4 | order-6 | $\boldsymbol{z}_{\nu=\mathrm{i}}^{(k)} * \boldsymbol{z}_{\xi}^{(k)}$ | $\boldsymbol{z}_{\nu=\mathrm{i}}^{(k)} * \boldsymbol{z}_{\mathrm{syn},\alpha\beta}^{(k)}$ | $\boldsymbol{z}_{\nu}^{(k)} * \boldsymbol{z}_{\mathrm{syn}}^{(k)}$ | others |
| 23 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $5.71_{\pm 5.71}$ | $0.05_{\pm 0.01}$ | $4.80_{\pm 0.96}$ | $47.07_{\pm 1.88}$ | $11.31_{\pm 1.76}$ | $39.80_{\pm 2.11}$ | $1.82_{\pm 1.82}$ |
| 71 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $0.00_{\pm 0.00}$ | $0.03_{\pm 0.00}$ | $5.02_{\pm 0.25}$ | $72.57_{\pm 0.70}$ | $4.00_{\pm 1.14}$ | $21.14_{\pm 2.14}$ | $2.29_{\pm 1.07}$ |
| 127 | $0.0_{\pm 0.0}$ | $1.50_{\pm 0.92}$ | $0.00_{\pm 0.00}$ | $0.26_{\pm 0.14}$ | $0.93_{\pm 0.18}$ | $82.96_{\pm 0.39}$ | $2.25_{\pm 0.64}$ | $14.13_{\pm 0.87}$ | $0.66_{\pm 0.66}$ |

100% of the per-freq
solutions are order-4/6

# Gradient Descent solutions matches with construction

| $d$ | %not | %non-factorable | | error ($\times 10^{-2}$) | | solution distribution (%) in factorable ones | | | |
|---|---|---|---|---|---|---|---|---|---|
| | order-4/6 | order-4 | order-6 | order-4 | order-6 | $\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\xi}$ | $\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\mathrm{syn},\alpha\beta}$ | $\boldsymbol{z}^{(k)}_{\nu} * \boldsymbol{z}^{(k)}_{\mathrm{syn}}$ | others |
| 23 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $5.71_{\pm 5.71}$ | $0.05_{\pm 0.01}$ | $4.80_{\pm 0.96}$ | $47.07_{\pm 1.88}$ | $11.31_{\pm 1.76}$ | $39.80_{\pm 2.11}$ | $1.82_{\pm 1.82}$ |
| 71 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $0.00_{\pm 0.00}$ | $0.03_{\pm 0.00}$ | $5.02_{\pm 0.25}$ | $72.57_{\pm 0.70}$ | $4.00_{\pm 1.14}$ | $21.14_{\pm 2.14}$ | $2.29_{\pm 1.07}$ |
| 127 | $0.0_{\pm 0.0}$ | $1.50_{\pm 0.92}$ | $0.00_{\pm 0.00}$ | $0.26_{\pm 0.14}$ | $0.93_{\pm 0.18}$ | $82.96_{\pm 0.39}$ | $2.25_{\pm 0.64}$ | $14.13_{\pm 0.87}$ | $0.66_{\pm 0.66}$ |

95% of the solutions are
factorizable into “2*3” or “2*2”

# Gradient Descent solutions matches with construction

| $d$ | %not | %non-factorable | | error ($\times 10^{-2}$) | | solution distribution (%) in factorable ones | | | |
|---|---|---|---|---|---|---|---|---|---|
| | order-4/6 | order-4 | order-6 | order-4 | order-6 | $z^{(k)}_{\nu=\mathrm{i}} * z^{(k)}_{\xi}$ | $z^{(k)}_{\nu=\mathrm{i}} * z^{(k)}_{\mathrm{syn},\alpha\beta}$ | $z^{(k)}_{\nu} * z^{(k)}_{\mathrm{syn}}$ | others |
| 23 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $5.71_{\pm 5.71}$ | $0.05_{\pm 0.01}$ | $4.80_{\pm 0.96}$ | $47.07_{\pm 1.88}$ | $11.31_{\pm 1.76}$ | $39.80_{\pm 2.11}$ | $1.82_{\pm 1.82}$ |
| 71 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $0.00_{\pm 0.00}$ | $0.03_{\pm 0.00}$ | $5.02_{\pm 0.25}$ | $72.57_{\pm 0.70}$ | $4.00_{\pm 1.14}$ | $21.14_{\pm 2.14}$ | $2.29_{\pm 1.07}$ |
| 127 | $0.0_{\pm 0.0}$ | $1.50_{\pm 0.92}$ | $0.00_{\pm 0.00}$ | $0.26_{\pm 0.14}$ | $0.93_{\pm 0.18}$ | $82.96_{\pm 0.39}$ | $2.25_{\pm 0.64}$ | $14.13_{\pm 0.87}$ | $0.66_{\pm 0.66}$ |

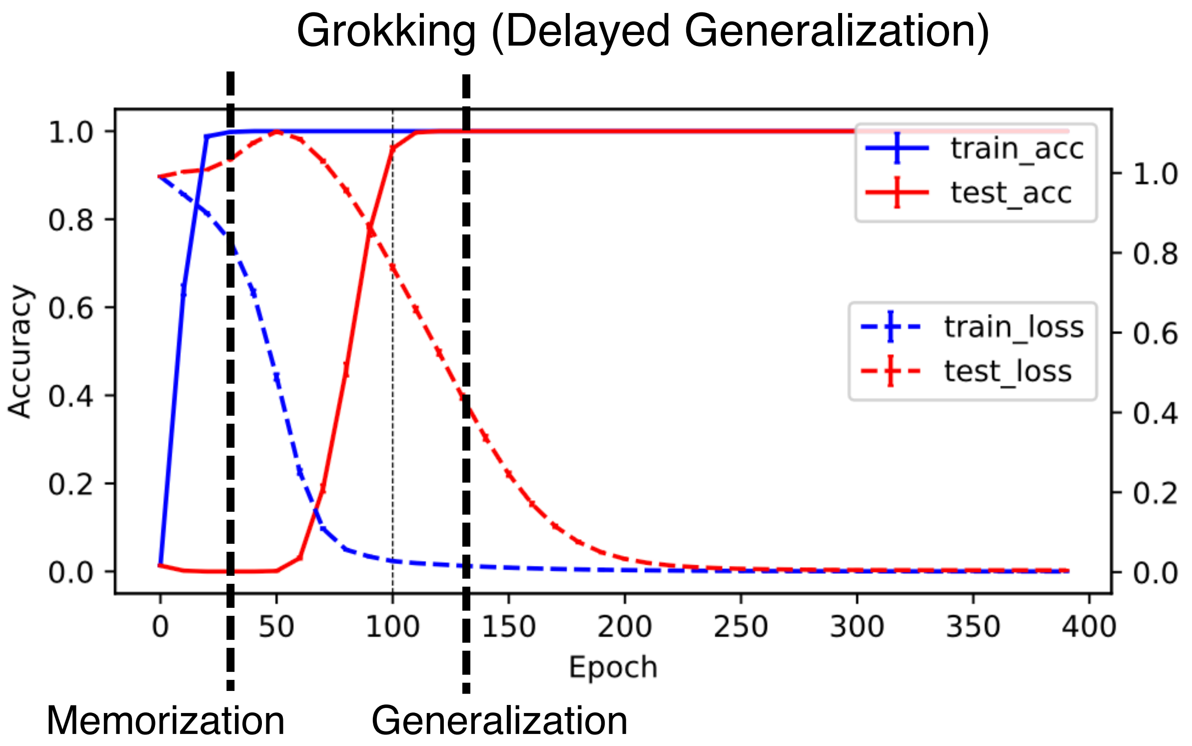
Factorization error is very small

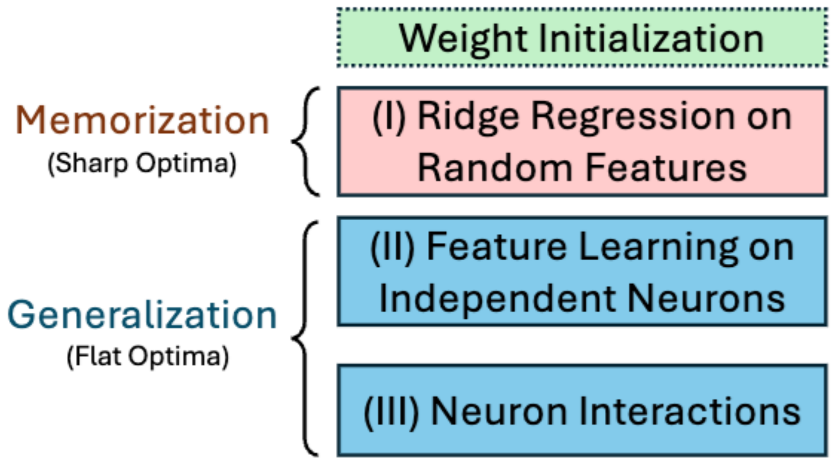
# Gradient Descent solutions matches with construction

| $d$ | %not | %non-factorable | | error ($\times 10^{-2}$) | | solution distribution (%) in factorable ones | | | |
|---|---|---|---|---|---|---|---|---|---|
| | order-4/6 | order-4 | order-6 | order-4 | order-6 | $\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\xi}$ | $\boldsymbol{z}^{(k)}_{\nu=\mathrm{i}} * \boldsymbol{z}^{(k)}_{\mathrm{syn},\alpha\beta}$ | $\boldsymbol{z}^{(k)}_{\nu} * \boldsymbol{z}^{(k)}_{\mathrm{syn}}$ | others |
| 23 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $5.71_{\pm 5.71}$ | $0.05_{\pm 0.01}$ | $4.80_{\pm 0.96}$ | $47.07_{\pm 1.88}$ | $11.31_{\pm 1.76}$ | $39.80_{\pm 2.11}$ | $1.82_{\pm 1.82}$ |
| 71 | $0.0_{\pm 0.0}$ | $0.00_{\pm 0.00}$ | $0.00_{\pm 0.00}$ | $0.03_{\pm 0.00}$ | $5.02_{\pm 0.25}$ | $72.57_{\pm 0.70}$ | $4.00_{\pm 1.14}$ | $21.14_{\pm 2.14}$ | $2.29_{\pm 1.07}$ |
| 127 | $0.0_{\pm 0.0}$ | $1.50_{\pm 0.92}$ | $0.00_{\pm 0.00}$ | $0.26_{\pm 0.14}$ | $0.93_{\pm 0.18}$ | $82.96_{\pm 0.39}$ | $2.25_{\pm 0.64}$ | $14.13_{\pm 0.87}$ | $0.66_{\pm 0.66}$ |

98% of the solutions can be factorizable into the constructed forms

# Understanding Grokking



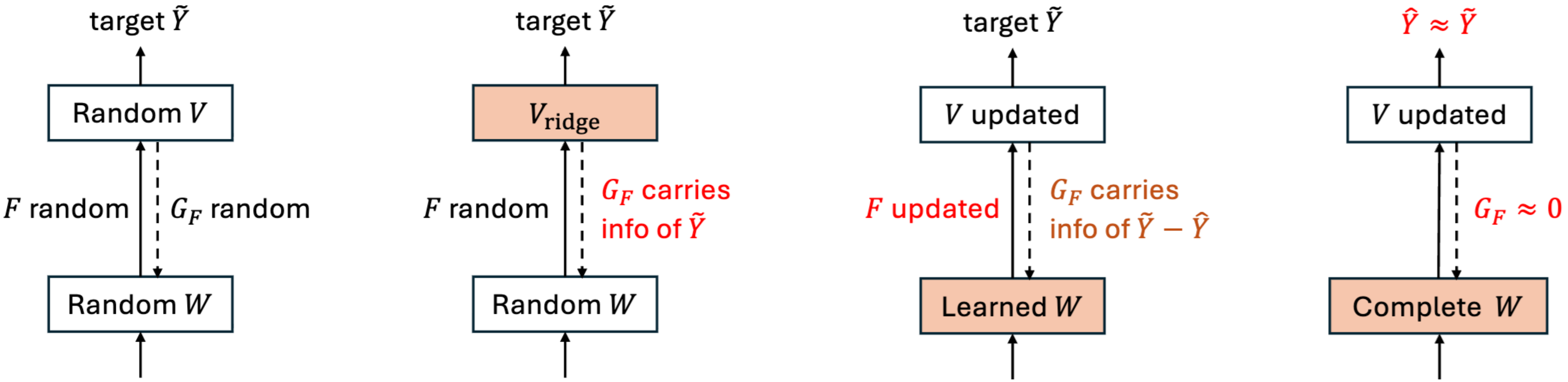






[Y. Tian, *Provable Scaling Laws of Feature Emergence from Learning Dynamics of Grokking*, arXiv'25]

# Stages of Grokking Behaviors

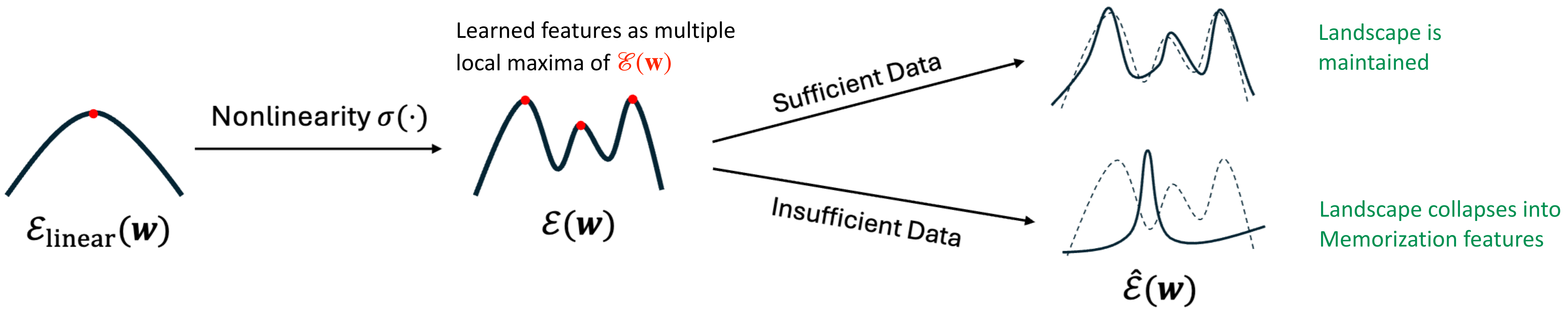
$$\hat{Y} = \sigma(XW)V$$




*(a) Initialization* *(b) After Lazy learning* *(c) After Independent Feature Learning* *(d) After Interactive Feature learning*

# Connect Emerging Features with Data / Architecture

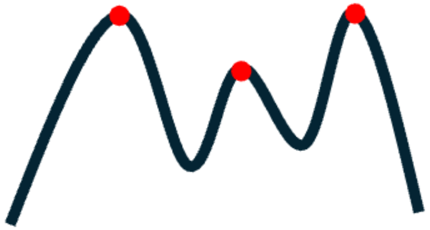
We discover that there exists an energy function $\mathcal{E}(\mathbf{w})$ that governs the feature learning process






[Y. Tian, *Provable Scaling Laws of Feature Emergence from Learning Dynamics of Grokking*, arXiv'25]

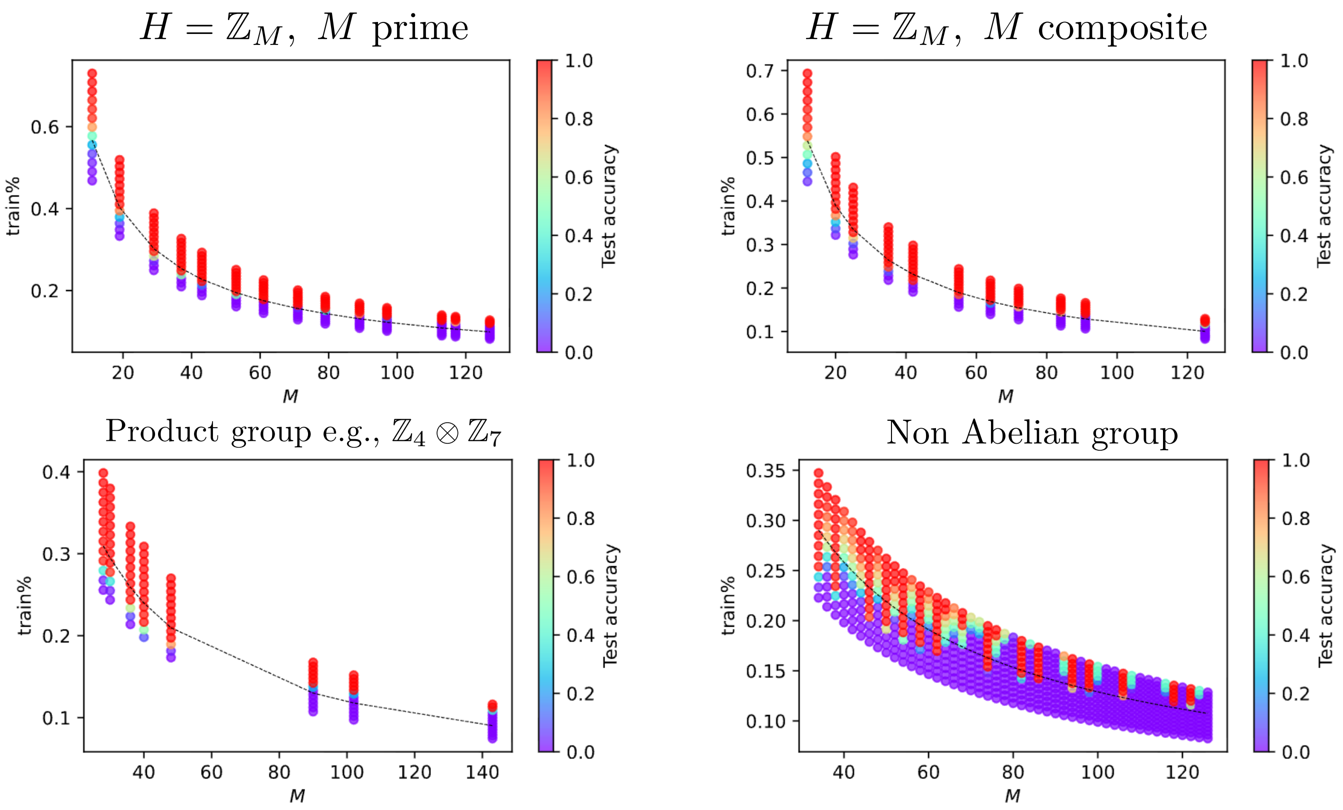
# Emerging Features are Symbolic!




$$\mathcal{E}(\mathbf{w}) = \frac{1}{2}\sum_h \langle \tilde{R}_h, S\rangle_F^2 = \frac{M}{2}\sum_{k\neq 0}\frac{1}{d_k}\left|\sum_r \mathrm{tr}(\hat{S}_{k,r})\right|^2$$

**Theorem 2** (Local maxima of $\mathcal{E}$ for group input). *For group arithmetics tasks with $\sigma(x) = x^2$, $\mathcal{E}$ has multiple local maxima $\mathbf{w}^* = [\mathbf{u}; \pm P\mathbf{u}]$. Either it is in a real irrep of dimension $d_k$ (with $\mathcal{E}^* = M/8d_k$ and $\mathbf{u} \in \mathcal{H}_k$), or in a pair of complex irrep of dimension $d_k$ (with $\mathcal{E}^* = M/16d_k$ and $\mathbf{u} \in \mathcal{H}_k \oplus \mathcal{H}_{\bar{k}}$). These local maxima are not connected. No other local maxima exist.*

# How much is **sufficient**? Provable Scaling Laws



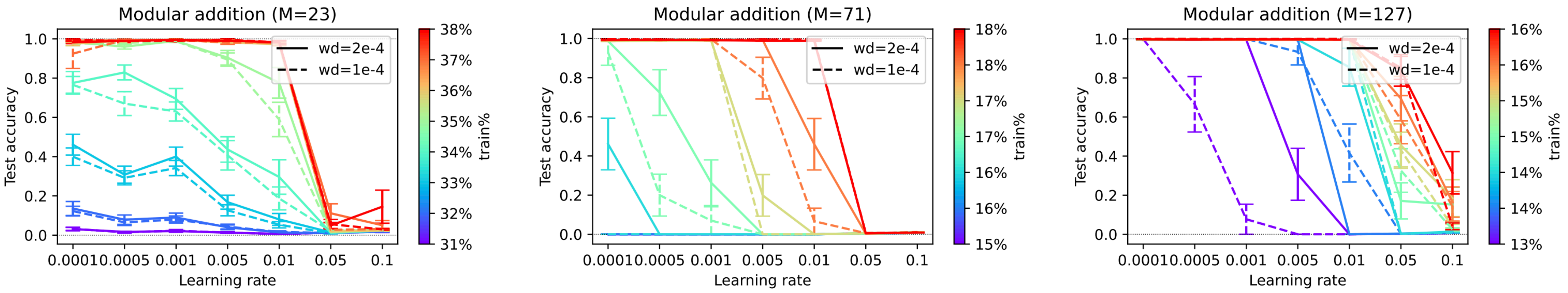

For Group Arithmetic tasks

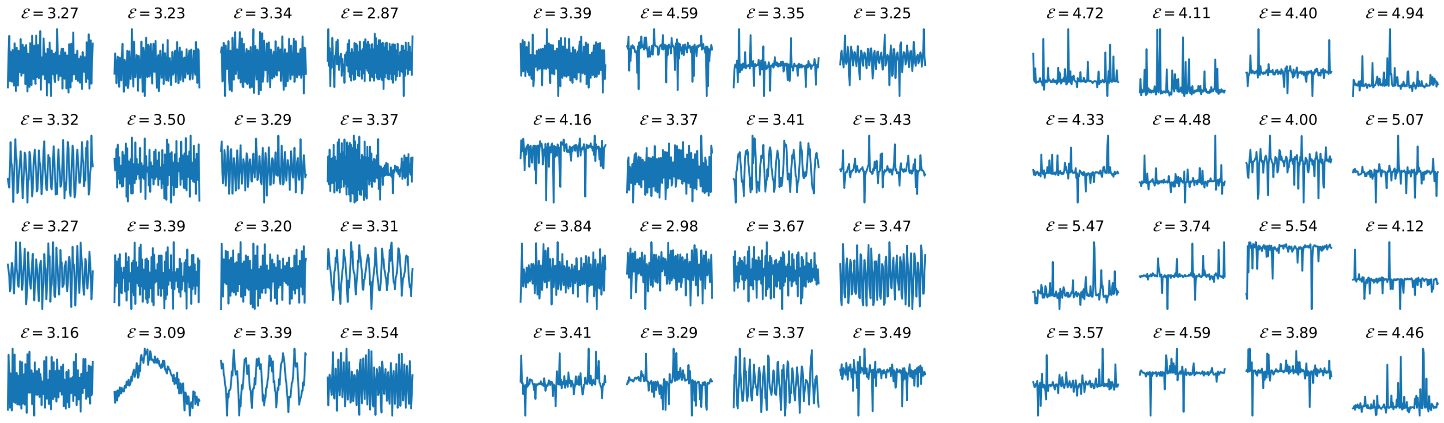
$$\text{Predict } h_1 h_2, \quad \text{given } h_1, h_2 \in H$$

*O(|H| log |H|)* data sample suffice to learn generalizable features

**Next Step:** Scale it to more complicated tasks and architectures

# Boundary between Memorization and Generalization



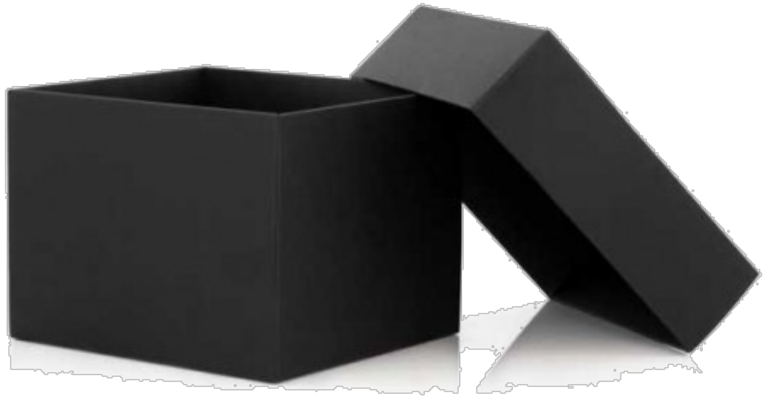



# Possible Implications

Do neural networks end up learning more efficient
**symbolic representations** that we don’t know?

Does gradient descent lead to a solution that
can be reached by **advanced algebraic operations**?

Will gradient descent become **obsolete**, eventually?



# Thanks!