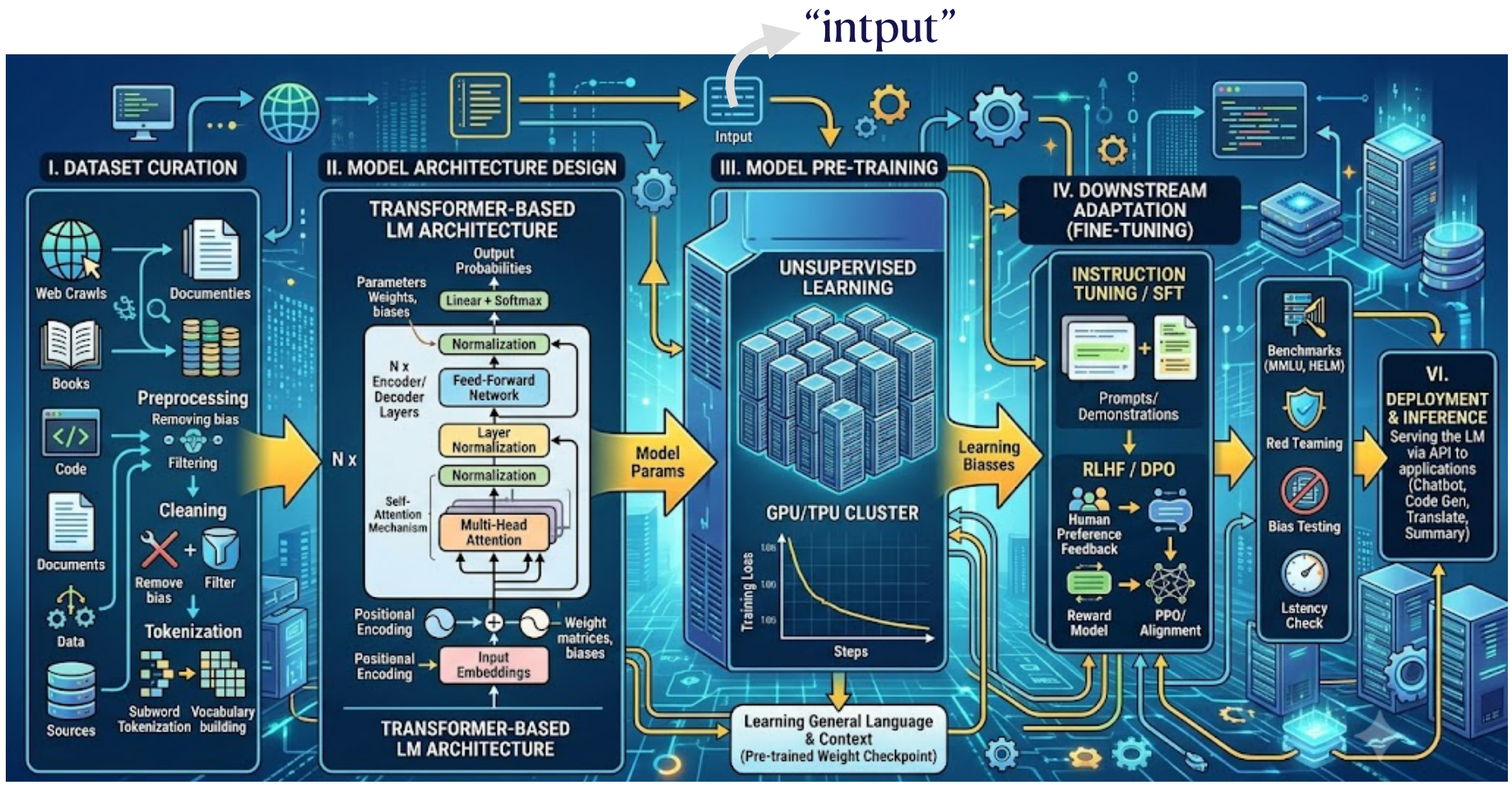
Improving reasoning efficiency through theoretically informed sandboxes

Bingbin Liu

Kempner Institute, Harvard University

Improving a complex system?

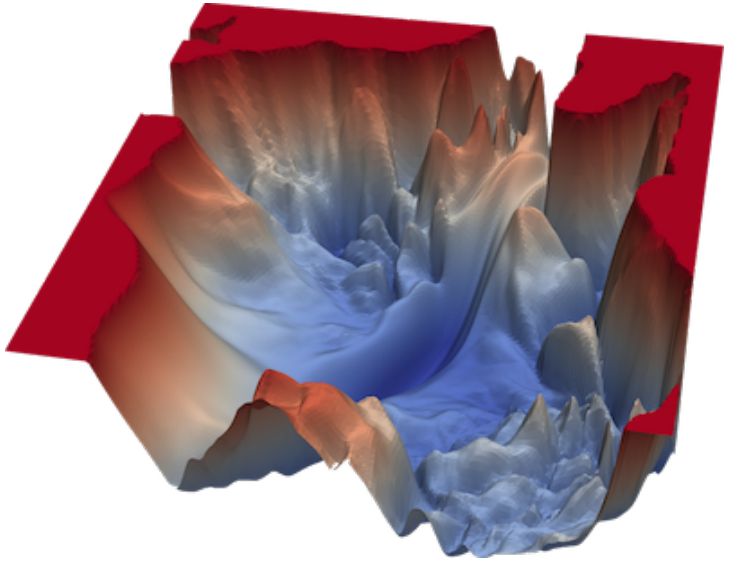


*PC Gemini

Improving a complex system? Sandbox!

**proper abstractions*

Reality (messy, complex)

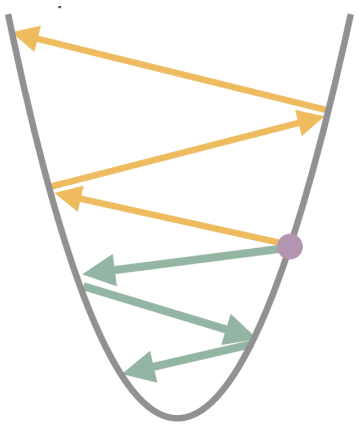


On VGGNet [Li et al. 17]

analyzable

reflective of practice

Sandbox (clean, simple)



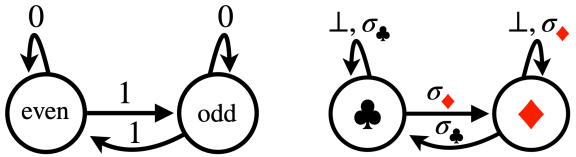
Informative for LLMs!

[Zhang et al. 25, Meterez et al. 26]

Efficient reasoning through sandboxes

Part 1: Compact reasoning **solutions**

automata



Parallel constructions + diagnosing models

Part 2: Improving **learning** efficiency

sparse parity

$$x = 1\ \underbrace{\text{-1 -1 1}}_{|S|=k}\ \text{-1 1 1 1 -1 1}$$

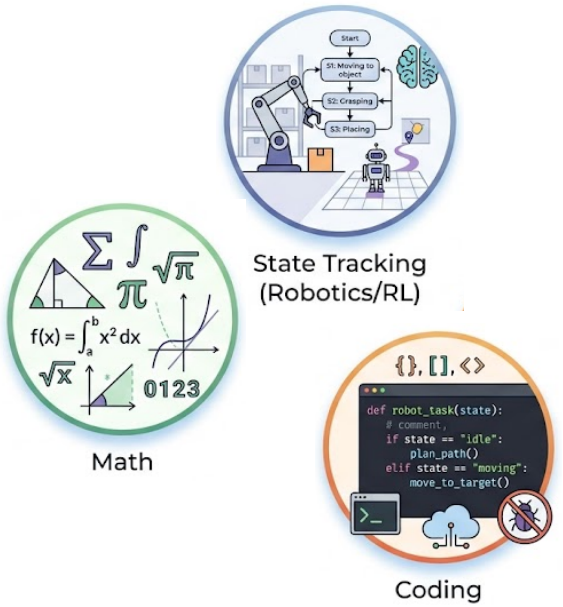
Data-related factors

Reflections

Part 1: Compact reasoning solutions

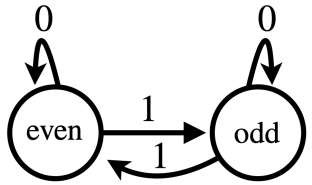
as computation

Sequential reasoning



diagnosing failures

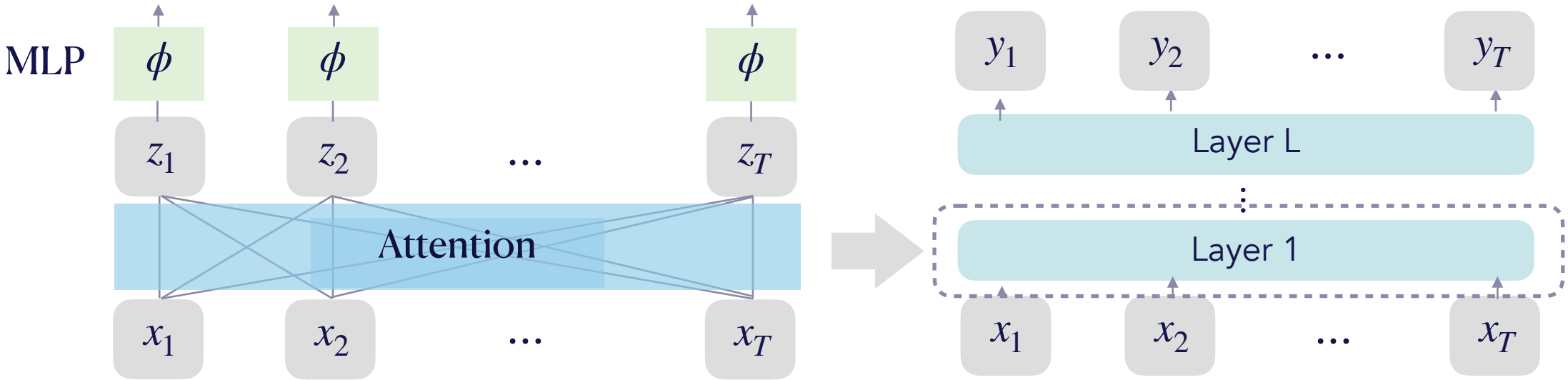
Automata (DFA)



Shallow, parallel solutions

Background: Transformer

In-parallel (across i) compute: 1) $z_i^{(l)} = \sum_j \alpha_{i,j}^{(l-1)} x_j^{(l-1)}$, 2) $x_i^{(l)} = \phi(z_i^{(l)})$.

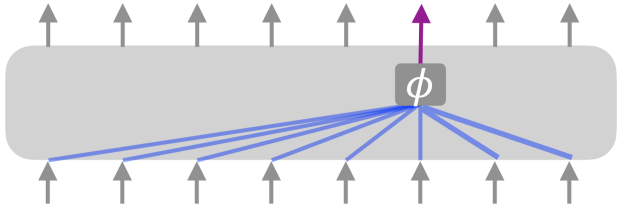


Background: Transformer

In-parallel (across i) compute: 1) $z_i^{(l)} = \sum_j \alpha_{i,j}^{(l-1)} x_j^{(l-1)}$, 2) $x_i^{(l)} = \phi(z_i^{(l)})$.

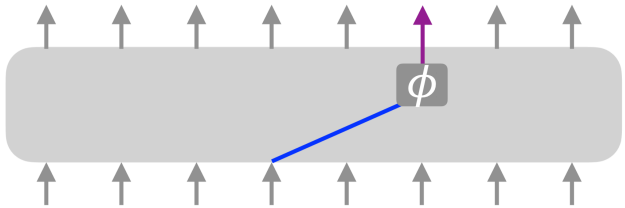
$$\sum_j \alpha_{i,j} = 1, \alpha_{i,j} \geq 0$$

1. Uniform attention: $\vec{\alpha}_i = [\frac{1}{T}, \frac{1}{T}, \cdots, \frac{1}{T}]$.



e.g. average, sum

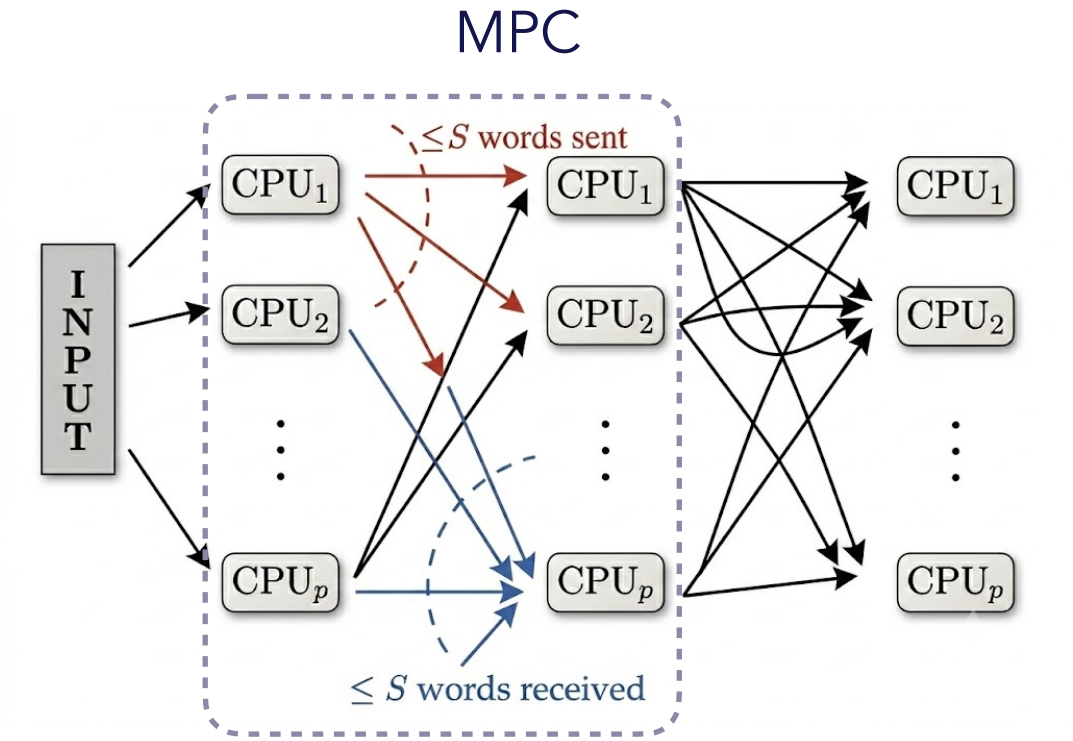
2. Sparse attention: $\vec{\alpha}_i = [0, \cdots, 0, 1, 0, \cdots]$.

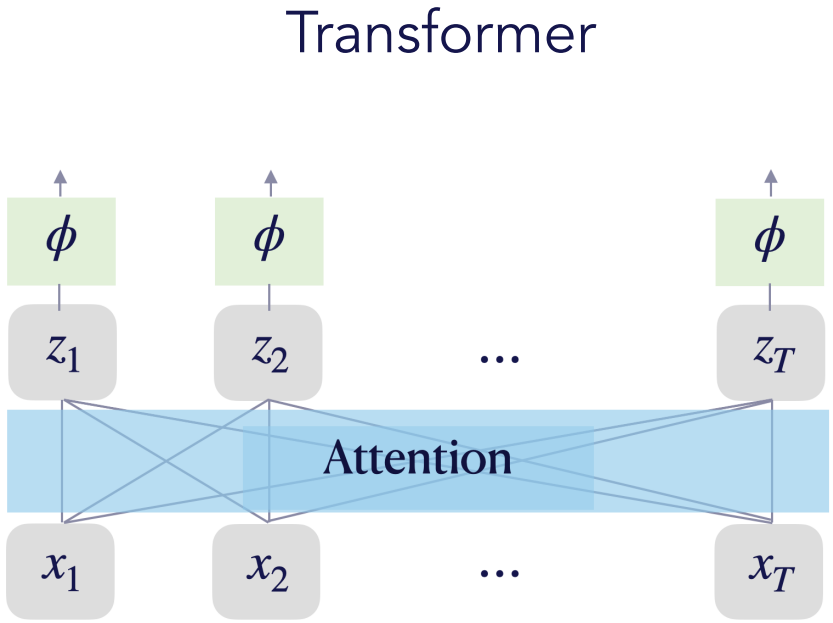


e.g. selection

Background: Transformer as parallel models

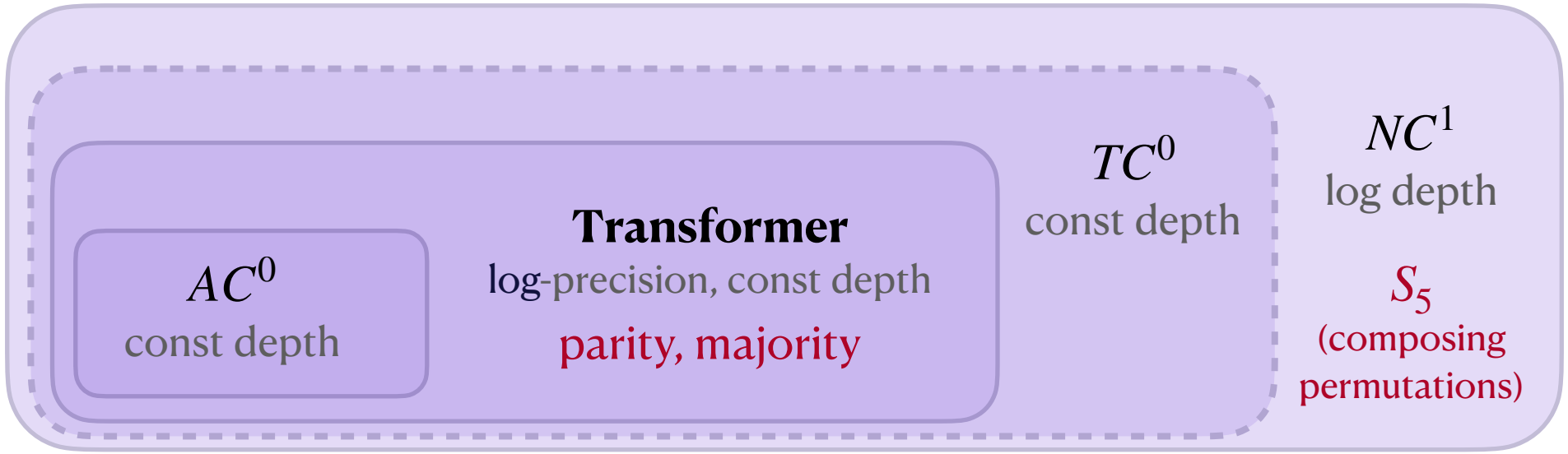
e.g. Massively Parallel Computation: positions/tokens as MPC nodes [Sanford et al. 24].





Background: Transformer as parallel models

e.g. circuit complexity: Transformer $\subset TC^0$ [Merrill & Sabhawal 22].



Parallel models for sequential reasoning?

automata

Sequential reasoning via automata

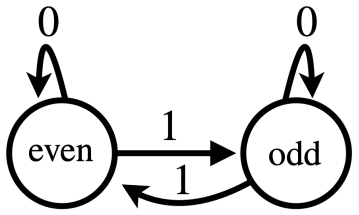
$\mathcal{A} := (Q, \Sigma, \delta)$ $\qquad q_t = \delta(\sigma_t, q_{t-1})$

states inputs transitions (Q is finite)

parity counter

$Q = \{\text{even}, \text{odd}\}$

$\Sigma = \{0, 1\}$

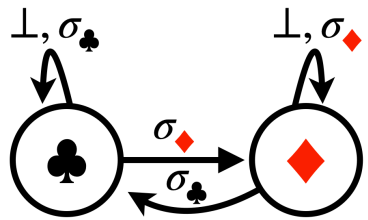


1-bit memory unit

$Q = \{\clubsuit, \diamondsuit\}$

$\Sigma = \{\sigma_\clubsuit, \sigma_\diamondsuit, \perp\}$

(no-op)

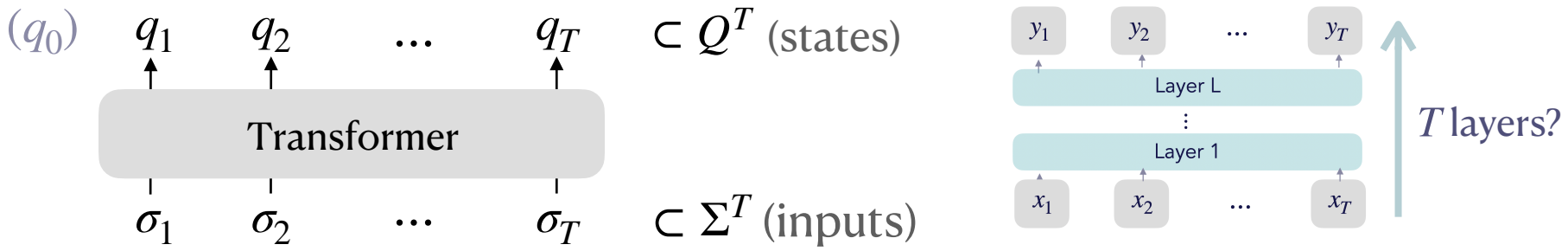


Sequential reasoning via automata

$$\mathscr{A} := (Q, \Sigma, \delta)$$
$$q_t = \delta(\sigma_t, q_{t-1})$$

Reasoning: simulating $\mathscr{A}$: learn a seq2seq function with length T.

(trivial with T steps)



1. Any $\mathscr{A}$: $O(\log T)$ layers

2. Solvable $\mathscr{A}$: $O(|Q|^2 \log |Q|)$ layers

Diagnosing & improving practices

Simulating T transitions in $\mathscr{A}$

$$\mathscr{A} := (Q, \Sigma, \delta)$$
$$q_t = \delta(\sigma_t, q_{t-1})$$

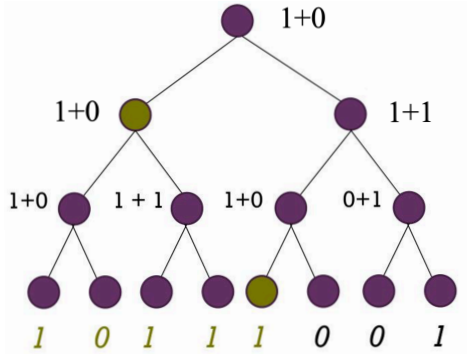
1. $O(\log T)$ layers for any $\mathscr{A}$: divide-and-conquer.

function composition

$$q_T = \delta(\sigma_T, \delta(\cdots, \delta(\sigma_2, \delta(\sigma_1, q_0)))) = (\underbrace{\delta(\sigma_T, \cdot)}_{\mathbb{R}^{|Q|\times|Q|}} \circ \cdots \circ \delta(\sigma_1, \cdot))(q_0)\,.$$

associativity

$$f_1 \circ f_2 \circ f_3 \circ f_4 = (f_1 \circ f_2) \circ (f_3 \circ f_4)$$



$\log T$ ***... shallower?***

$\Omega(\log T)$, *assuming* $TC^0 \neq NC^1$. [Merrill & Sabhawal 22]

- aka. "associative scan" in **prefix sum** [Blelloch 93], subquadratic models e.g. **Mamba** [Gu & Dao 23].

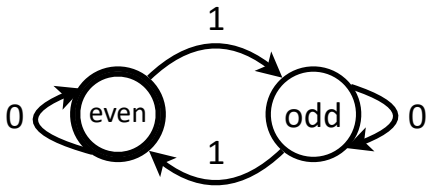
Simulating T transitions in $\mathscr{A}$

$$\mathscr{A} := (Q, \Sigma, \delta)$$
$$q_t = \delta(\sigma_t, q_{t-1})$$

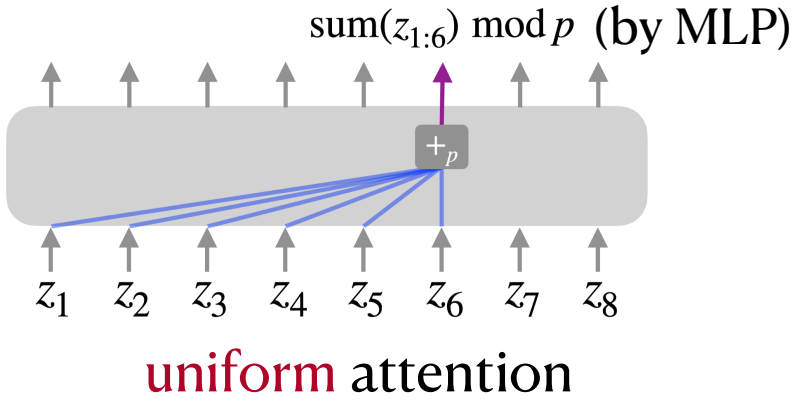
2. $O(|Q|^2 \log |Q|)$ layers for solvable $\mathscr{A}$.

- Idea: certain compositions are highly parallelizable.
  $O(1)$ layers if ***commutative***

e.g. parity:



$$q_T = \left(\sum_{t \in [T]} x_i\right) \mod 2$$



Simulating T transitions in $\mathscr{A}$

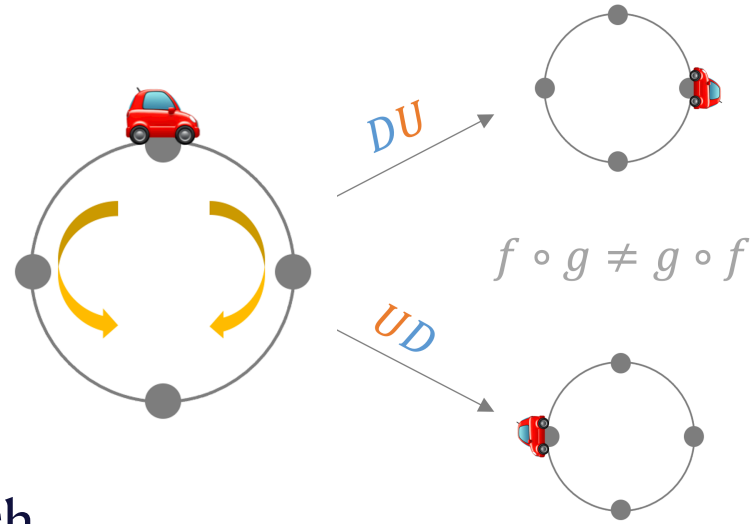
$$\mathscr{A} := (Q, \Sigma, \delta)$$
$$q_t = \delta(\sigma_t, q_{t-1})$$

2. $O(|Q|^2 \log |Q|)$ layers for solvable $\mathscr{A}$.

- **Krohn-Rhodes** decomposition: factorize into parallelizable "base cases".

$Q = \{$ 🚗 , 🚗 $\} \times \{0,1,2,3\}, \Sigma = \{D$ (drive), U (U-turn)$\}$.

$q_0 = \{$🚗$,0\}, \sigma_{1:T} = DDDUDDUUD \rightarrow q_T$?



1. Direction: parity of U.
2. Position: signed sum of D (mod 4).

$O(1)$ layer each.

Simulating T transitions in $\mathscr{A}$

$$\mathscr{A} := (Q, \Sigma, \delta)$$
$$q_t = \delta(\sigma_t, q_{t-1})$$

2. $O(|Q|^2 \log |Q|)$ layers for solvable $\mathscr{A}$.

- **Krohn-Rhodes** decomposition: factorize into parallelizable "base cases".

Transformation (semi)group $\mathscr{T}(\mathscr{A}) := \{\delta(\sigma, \cdot) : \sigma \in \Sigma\}$ under composition.

Recall: $q_T = (\delta(\sigma_T, \cdot) \circ \cdots \circ \delta(\sigma_1, \cdot))(q_0)$

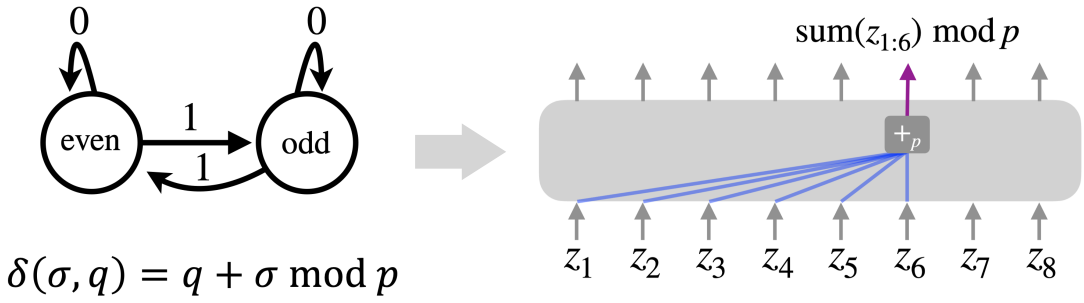
Simulating T transitions in $\mathscr{A}$

$$\mathscr{A} := (Q, \Sigma, \delta)$$
$$q_t = \delta(\sigma_t, q_{t-1})$$

2. $O(|Q|^2 \log |Q|)$ layers for solvable $\mathscr{A}$.

- **Krohn-Rhodes** decomposition: factorize into parallelizable "base cases".

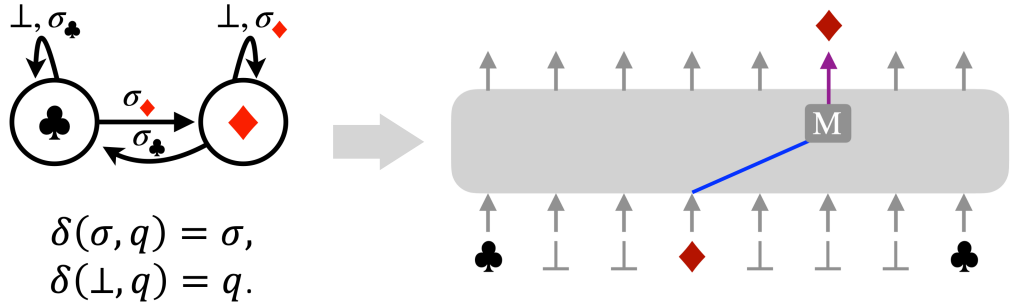
***2 cases**, each representable by 1 Transformer layer.*



$\delta(\sigma, q) = q + \sigma \bmod p$

mod(-p) counter

uniform attention



$\delta(\sigma, q) = \sigma,$
$\delta(\perp, q) = q.$

memory unit

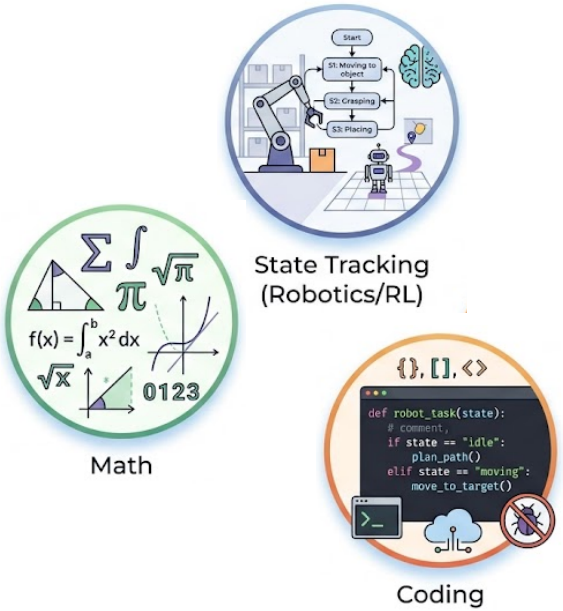
1-sparse attention

e.g. count, sum, average

e.g. selection, long-range, induction head.

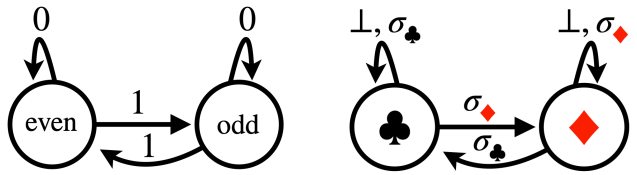
Part 1: Compact reasoning solutions

Sequential reasoning



Diagnose failures
in generalization

Automata (DFA)

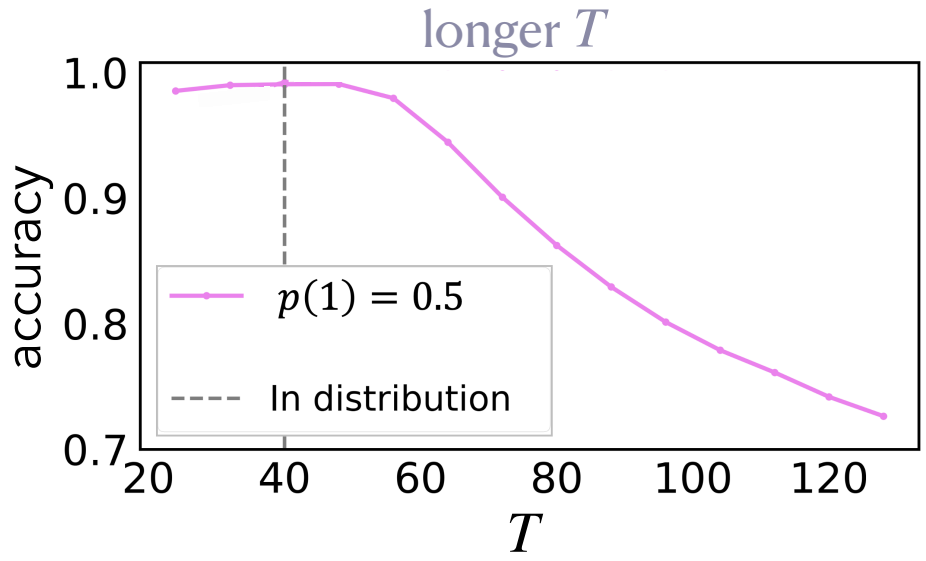


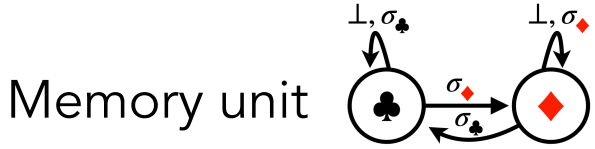
Any $\mathscr{A}$: $O(\log T)$ depth

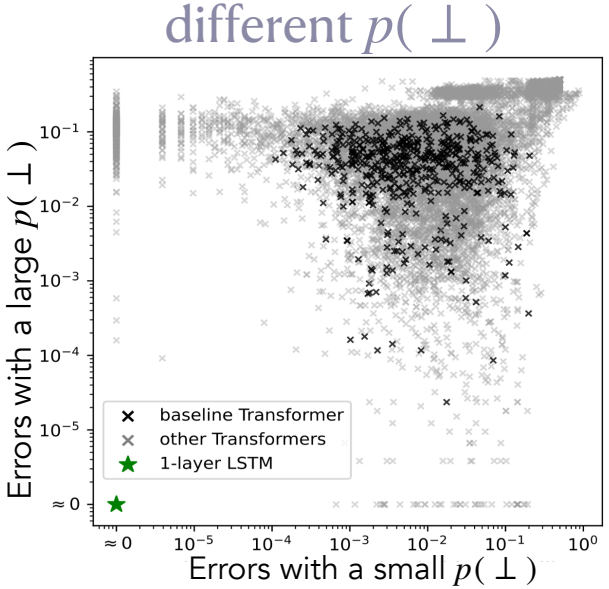
Solvable $\mathscr{A}$: $O(|Q|^2 \log |Q|)$ depth

Diagnosing models using “base cases”

Generalization out-of-domain/distribution.



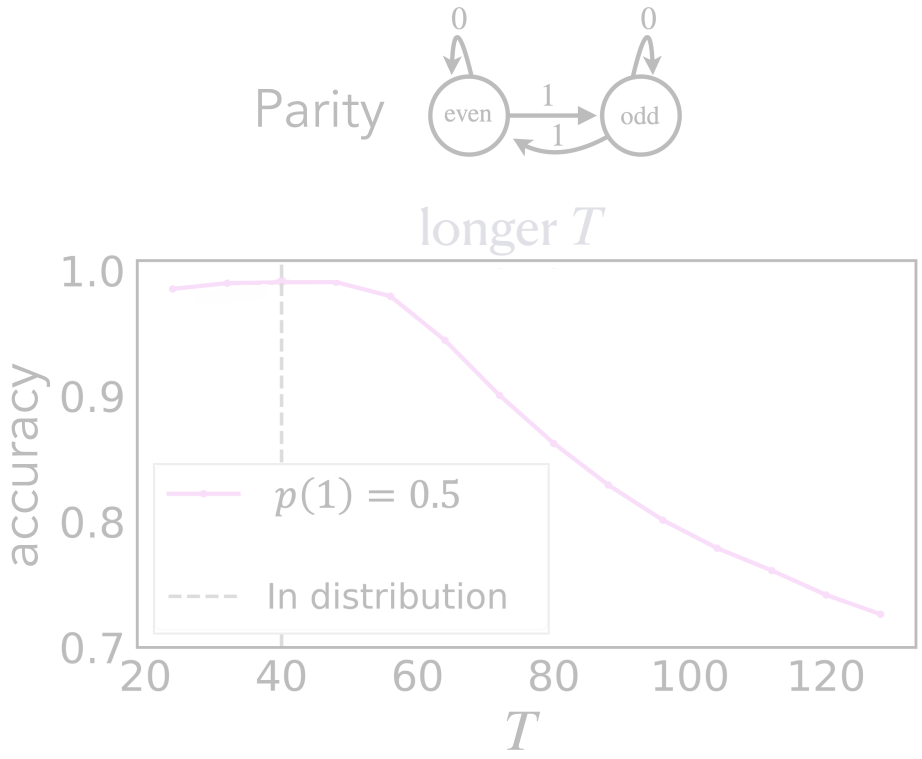


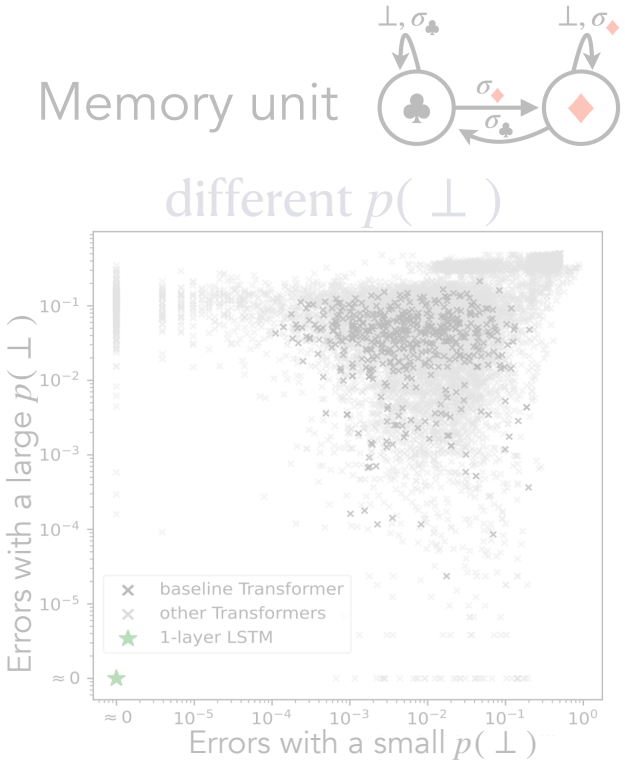


LAGKZ: Transformers Learn Shortcuts to Automata. ICLR 23.
LAGKZ: Exposing Attention Glitches with Flip-Flop Language Modeling. NeurIPS 23.

Diagnosing models using “base cases”

Generalization: errors due to intrinsic architectural challenges. *... easy for RNNs.*



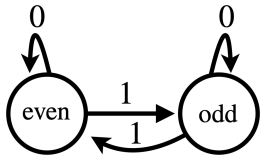


LAGKZ: Transformers Learn Shortcuts to Automata. ICLR 23.
LAGKZ: Exposing Attention Glitches with Flip-Flop Language Modeling. NeurIPS 23.

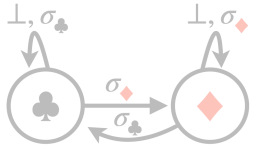
Diagnosing models using "base cases"

Generalization: errors due to intrinsic architectural challenges.

Parity



Memory unit

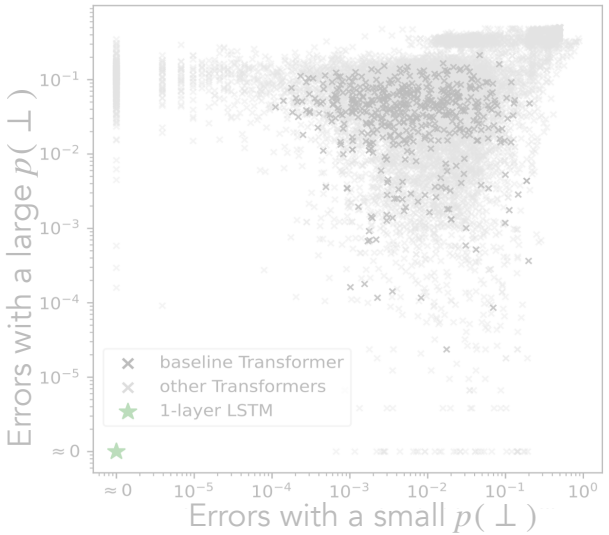


MLP: no nonlinear generalization.

$$q_T = (\sum_i x_i) \mod 2$$

memorized

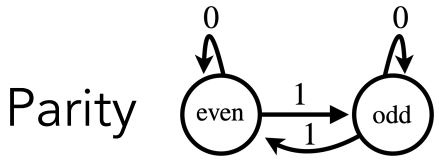
Wrong output on unseen #1s [Xu et al. 20]



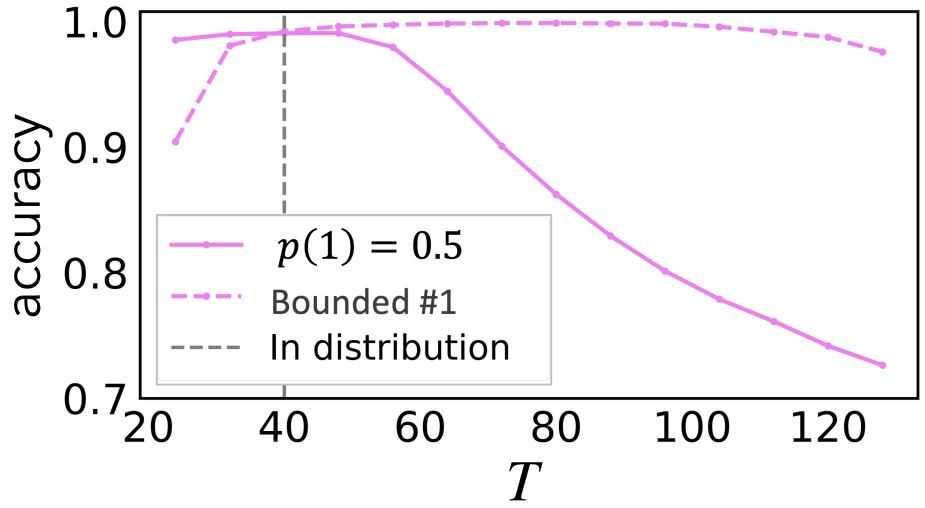
LAGKZ: Transformers Learn Shortcuts to Automata. ICLR 23.
LAGKZ: Exposing Attention Glitches with Flip-Flop Language Modeling. NeurIPS 23.

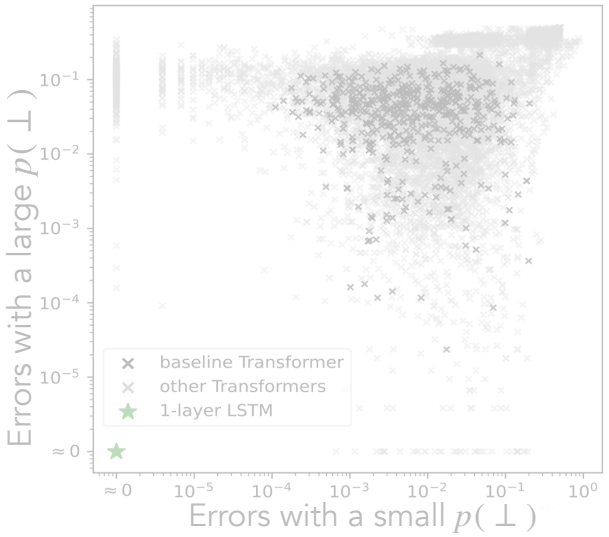
Diagnosing models using “base cases”

Generalization: errors due to intrinsic architectural challenges.



MLP: no nonlinear generalization.

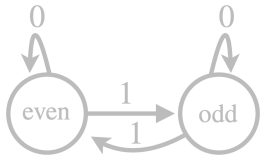




LAGKZ: Transformers Learn Shortcuts to Automata. ICLR 23.
LAGKZ: Exposing Attention Glitches with Flip-Flop Language Modeling. NeurIPS 23.

Diagnosing models using "base cases"

Generalization: errors due to intrinsic architectural challenges.

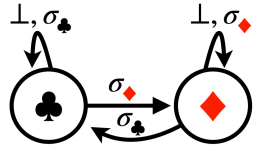
Parity 

MLP: no nonlinear generalization.

$$q_T = (\sum_i x_i) \mod 2$$

memorized

Wrong output on unseen #1s [Xu et al. 20]

Memory unit 

Attention: "diluted" weights.

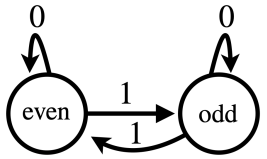
$$z_i^{(l)} = \sum_j \alpha_{i,j}^{(l-1)} x_j^{(l-1)}, \sum_j \alpha_{i,j} = 1, \alpha_{i,j} \geq 0$$

$$\alpha_{\max} = \frac{\exp(q^\top k_{\max})}{\cdots + \exp(q^\top k_{\max})}$$

Fixing models using "base cases"

Generalization: errors due to intrinsic architectural challenges.

Parity



MLP: no nonlinear generalization.

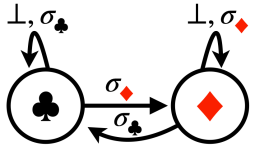
$$q_T = \left(\sum_i x_i\right) \mod 2$$

memorized

Wrong output on unseen #1s [Xu et al. 20]

Fix: data exposure, e.g. priming [Jelassi et al. 23].

Memory unit



Attention: "diluted" weights.

$$z_i^{(l)} = \sum_j \alpha_{i,j}^{(l-1)} x_j^{(l-1)}, \sum_j \alpha_{i,j} = 1, \alpha_{i,j} \geq 0$$

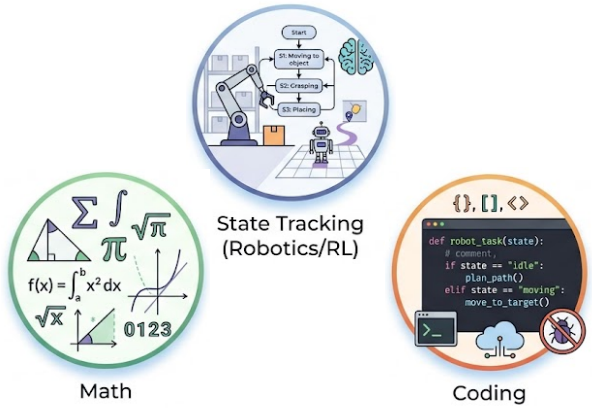
$$\alpha_{\max} = \frac{\exp(q^\top k_{\max})}{\cdots + \exp(q^\top k_{\max})}$$

Fix: temperature; positional enc.

[Chiang & Cholak 22] [Yang et al. 25]

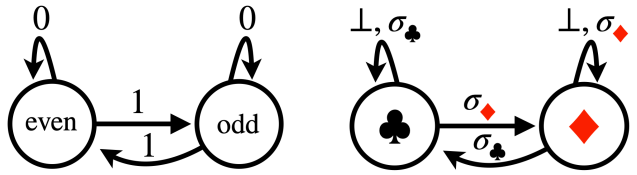
Part 1: Compact reasoning solutions

Sequential reasoning



Diagnose failures
in generalization

Automata (DFA)



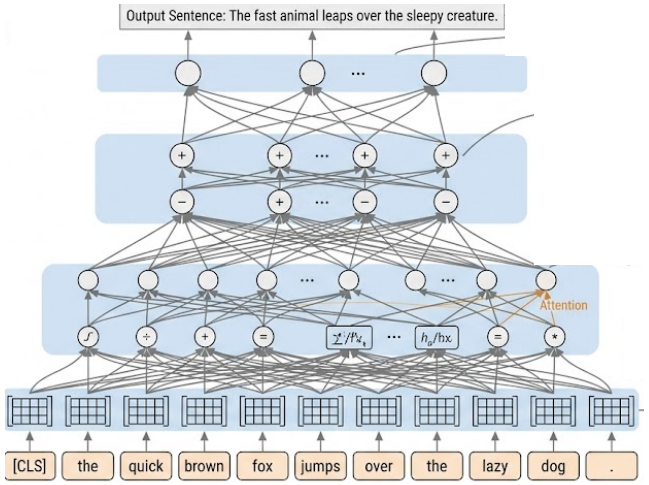
$o(T)$ layers

(Approximate) compact solutions exist. How to learn them efficiently?

Part 2: Efficient learning

Feature learning

in large models



Sparse parity

in 2-layer nets

$$x \in \{\pm 1\}^d, y = \prod_{i \in S} x_i, \ |S| = k$$

e.g. $d = 6, k = 3$

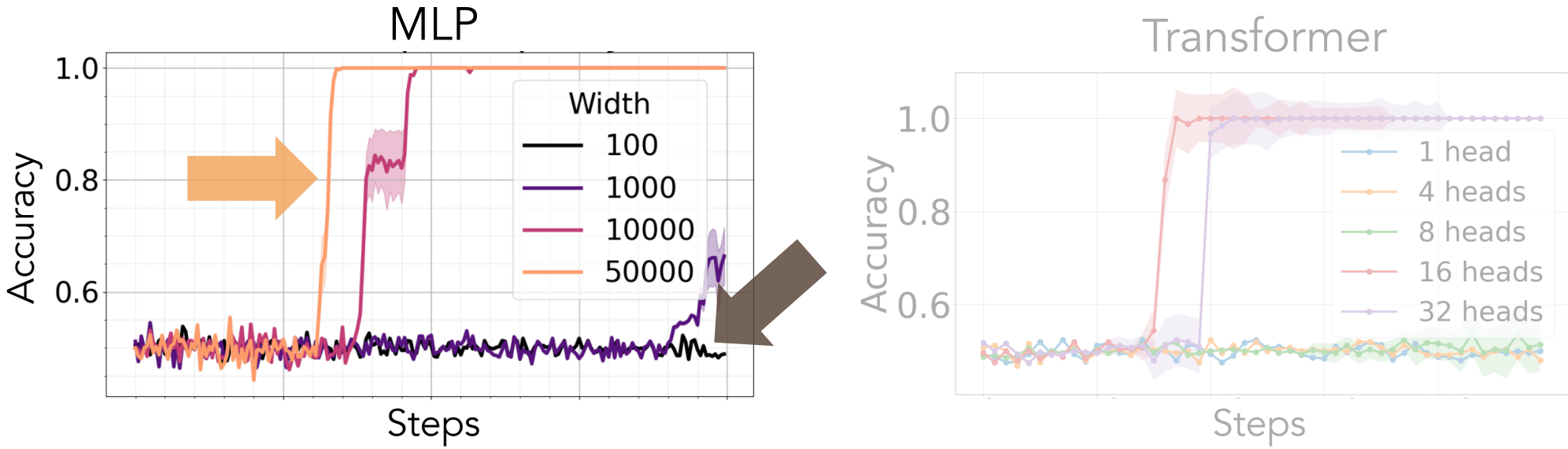
$x = [1 \ \underbrace{-1 \ -1 \ 1}_{S} \ -1 \ 1]$

$y = 1$

Learning sparse parity is challenging

Idea: 1) SGD as a statistical query (SQ) algo; 2) parity is SQ hard. [Kearns 98, FGV17, AKMSS21, EGKMZ23]

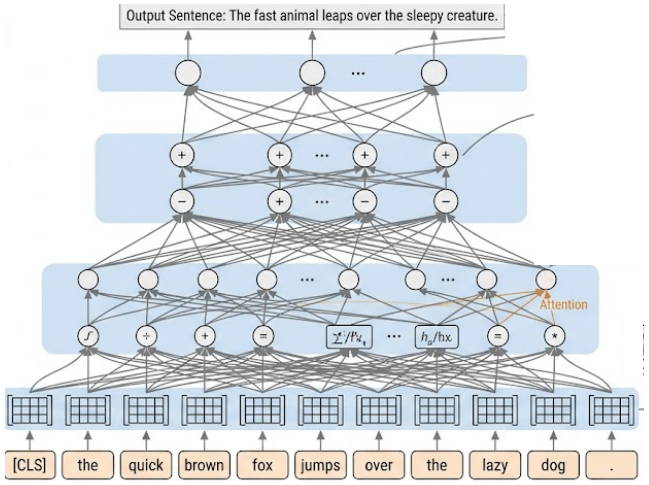
Implication: Smaller models learn more slowly.



Part 2: Efficient learning

Feature learning

in large models



improve LLMs

Sparse parity

in 2-layer nets

$$x \in \{\pm 1\}^d, y = \prod_{i \in S} x_i, \ |S| = k$$

Change **data**-related factors

1. Richer supervision via **distillation**.
2. Data **repetition**

Improving learning with distillation

Distillation: train small models better, by leveraging powerful pretrained models.
"student" "teacher"

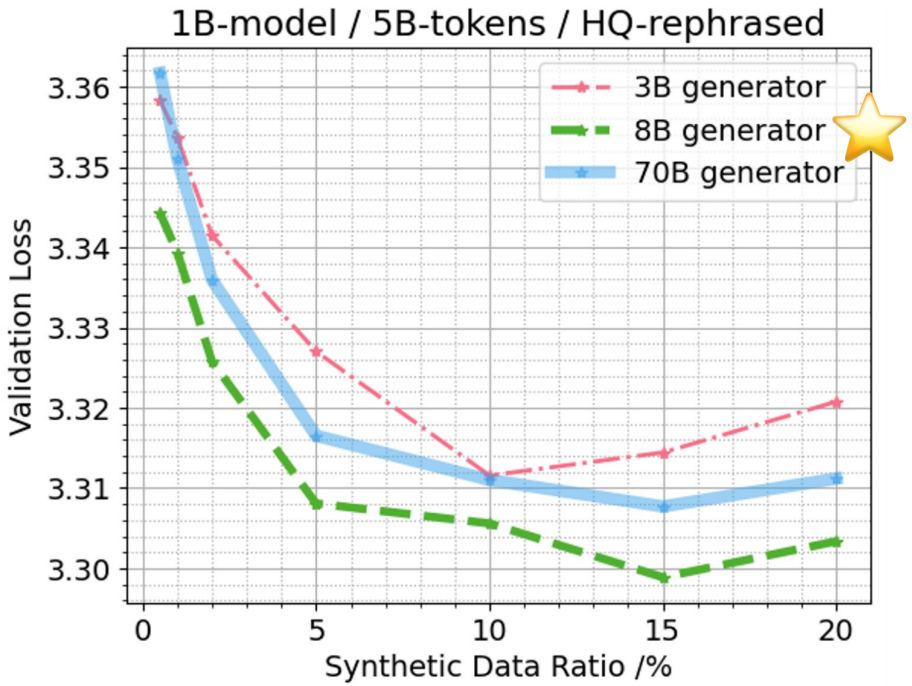
- Student learns from teacher's output:

Supervised learning: $\ell(f(x), y)$ Distillation from f_T: $\ell(f(x), f_T(x))$

Model	AIME 2024		MATH-500	GPQA Diamond	LiveCodeBench
	pass@1	cons@64	pass@1	pass@1	pass@1
QwQ-32B-Preview	50.0	60.0	90.6	54.5	41.9
DeepSeek-R1-Zero-Qwen-32B	47.0	60.0	91.6	55.0	40.2
DeepSeek-R1-Distill-Qwen-32B	**72.6**	**83.3**	**94.3**	**62.1**	**57.2**

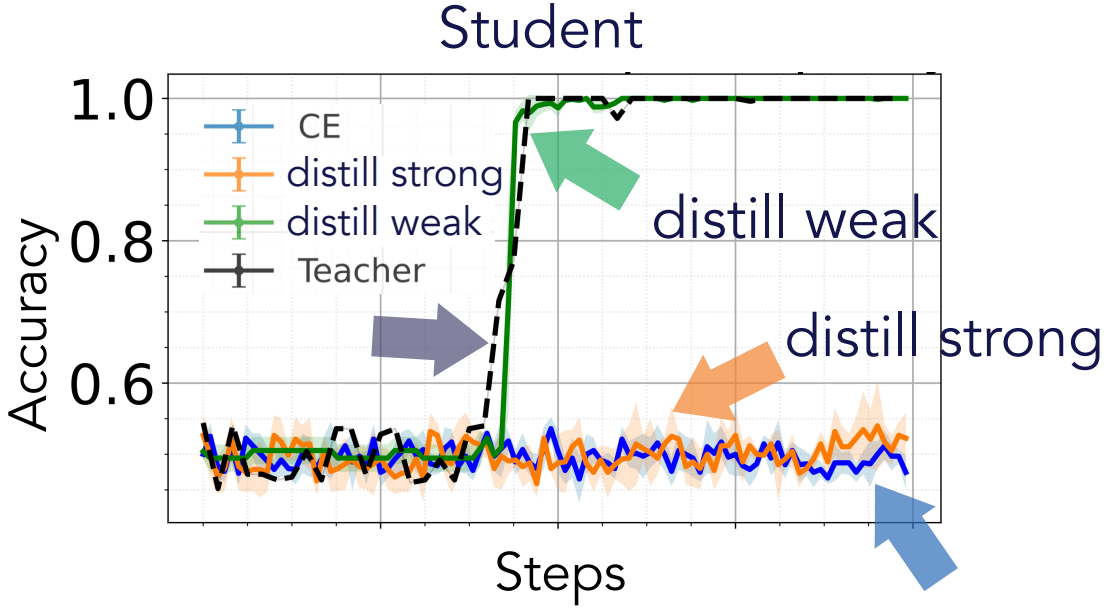
[DeepSeek R1 report]

Stronger model ≠ better teacher [Mirzadeh et al. 19]



[Kang et al. 25]

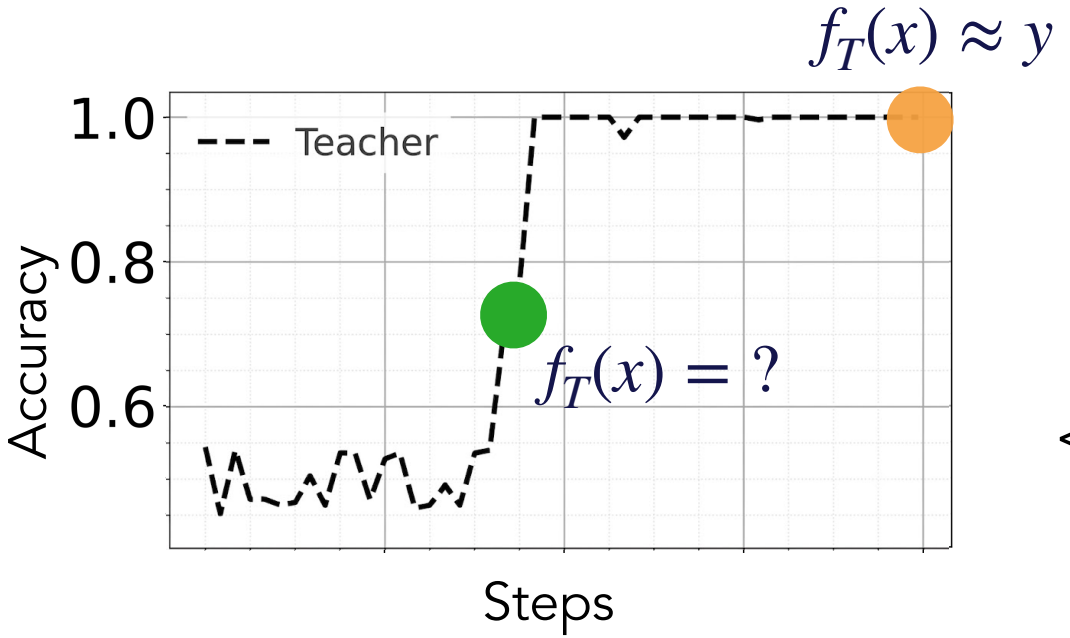
Sparse parity

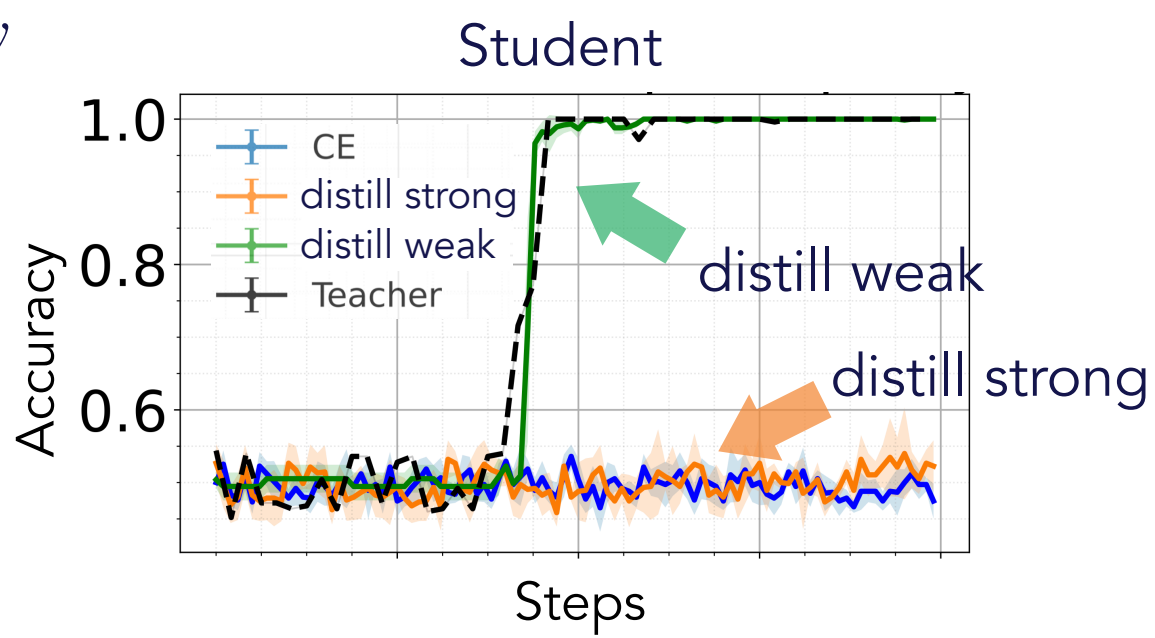


Which teachers?

Stronger vs weaker teachers

Different checkpoints (i.e. diff steps during training).

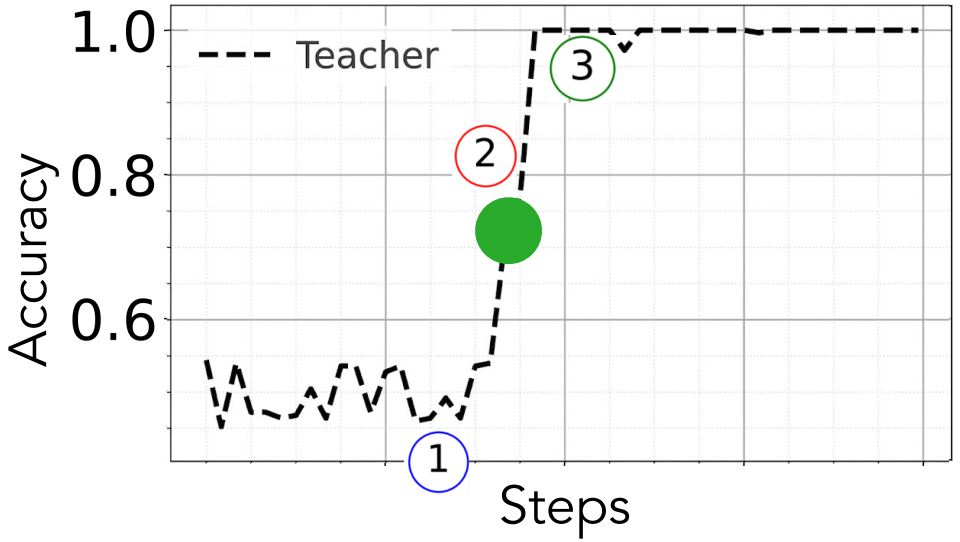




Why is the weaker teacher better?

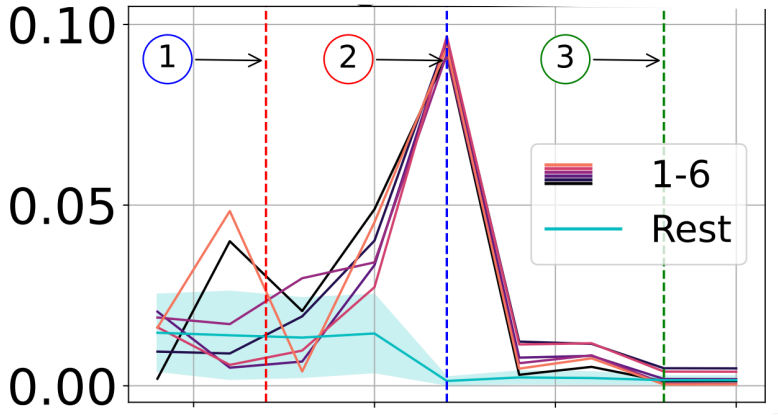
Intermediate checkpoints provides easier-to-learn "subtasks". ... **reveals** S.

during phase transition



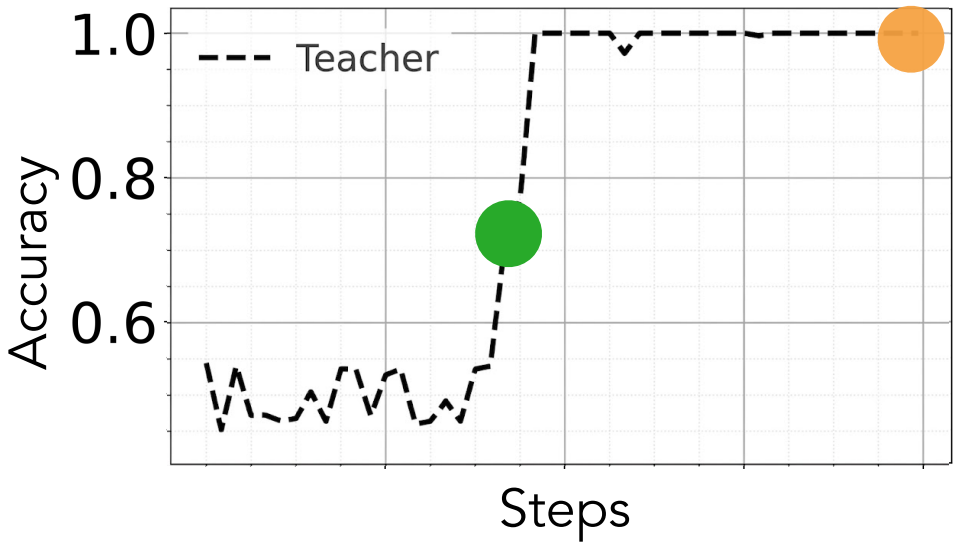
$\chi_i := \langle x_i, f_T \rangle$

larger $|\chi_{\{i\}}|$ for $i \in S$



Why is the weaker teacher better?

Intermediate checkpoints provides easier-to-learn "subtasks". ... 🟢 **reveals** S.



$$f_T \approx \sum_{i \in S} c_i x_i \rightarrow \text{learnable in } \tilde{\Theta}_{k,\epsilon}(d^2) \text{ steps.}$$

$$y = \prod_{i \in S} x_i \rightarrow \text{learnable in } \Omega(d^k) \text{ steps.}$$

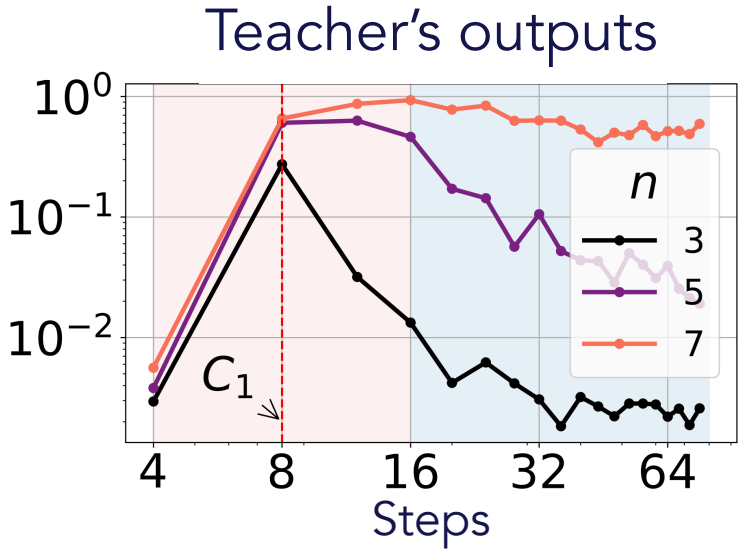
Fewer steps to learn *lower-degree* polynomials

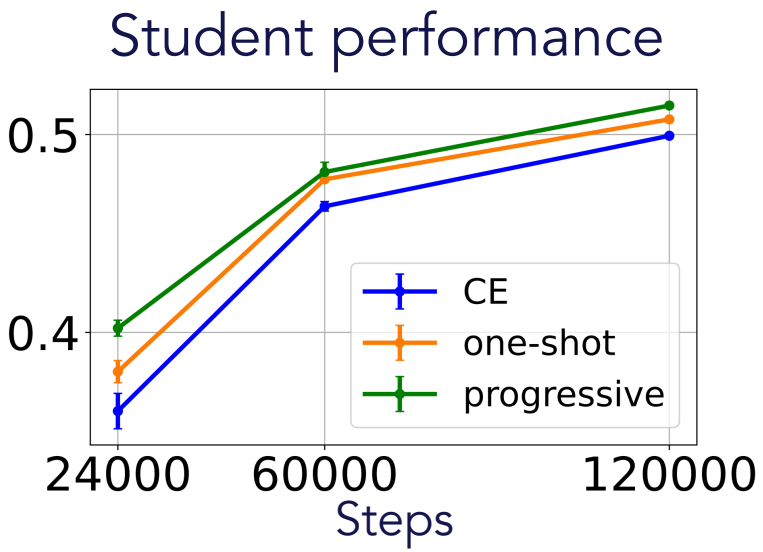
[Edelman et al. 22, Abbe et al. 23]

Why is the weaker teacher better?

Intermediate checkpoints provides easier-to-learn "subtasks".

Same observations for language models





Sensitivity *(c.f. Vatsal's talk)*

Distillation to accelerate learning

TL;DR: teacher choice matters. ... weaker / student-dependent.

A lot more to gain!

Practically impactful.

- Compute-efficient training?
  e.g. on-policy distillation [Qwen 3, TML's blog]
- Better sub-quadratic-time models?
  e.g. initialization [Bick et al. 24, Wang et al. 24]

Conceptually interesting

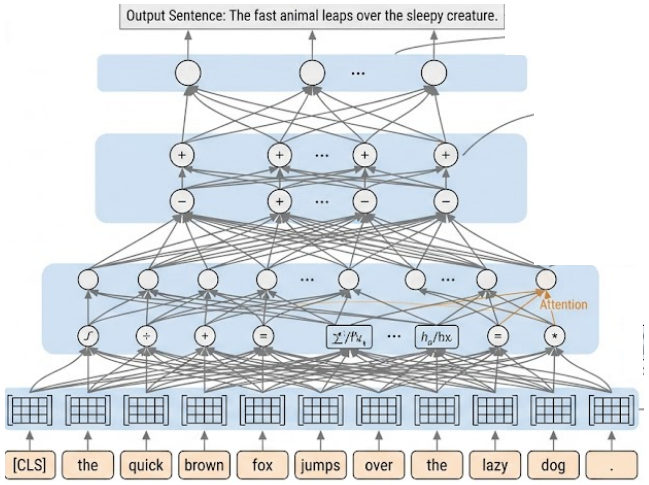
- Learning with dense supervision / CoT
  [Joshi et al. 25, Kim et al. 25]
- Imitation learning [Rohatgi et al. 25]
- Model stealing [Liu & Moitra 24]

P*L*MRG: Progressive distillation induces an implicit curriculum. ICLR 25.

Part 2: Efficient learning

Feature learning

in large models



improve LLMs

Sparse parity

in 2-layer nets

$$x \in \{\pm 1\}^d, y = \prod_{i \in S} x_i, \ |S| = k$$

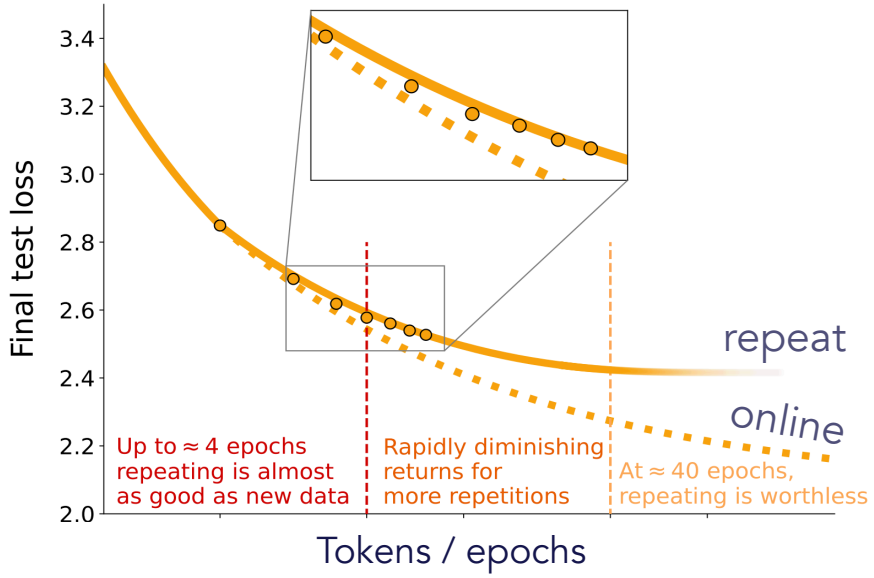
Change **data**-related factors

1. Richer supervision via **distillation**.
2. Data **repetition**

Effect of data repetition

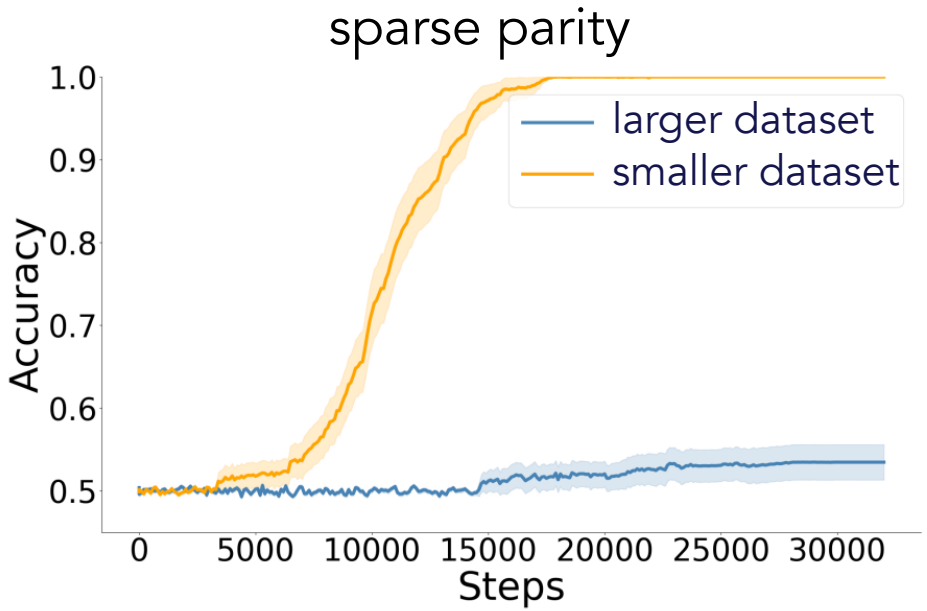
*New regime: **limited data**, infinite compute*

... worse test loss.



Muennighoff et al. 23

... or *better* reasoning accuracy?



Muennighoff et al.: Scaling Data-Constrained Language Models. NeurIPS 23.

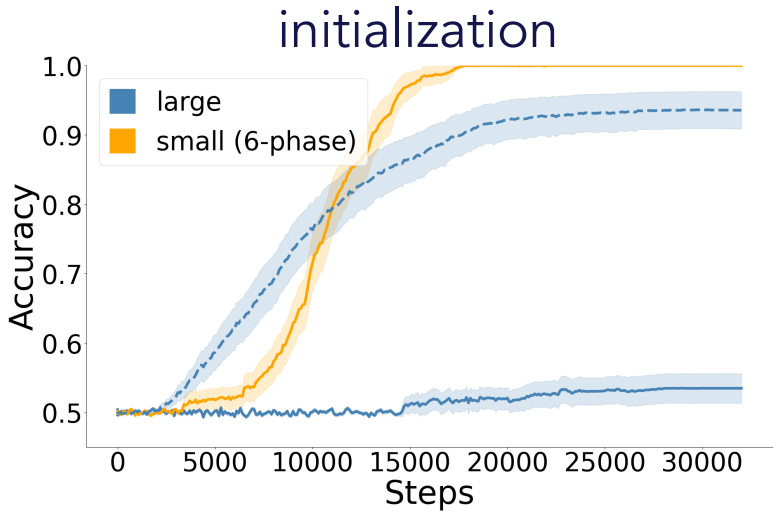
LEGL: Less Data, Faster Training: repeating smaller datasets speeds up learning via sampling biases. ICML 26.

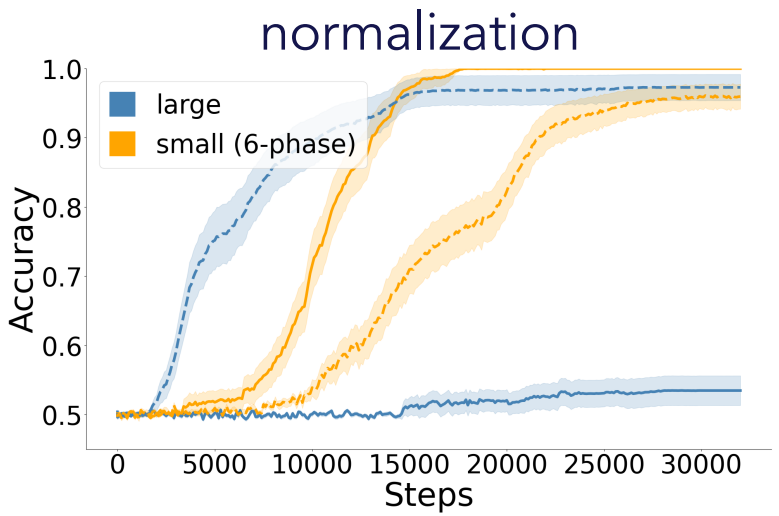
Effect of data repetition

*New regime: **limited data**, infinite compute*

Coupled effect: improved data efficiency can lead to improved compute efficiency.

Data repetition as ***favorable optimization bias***.





LEGL: Less Data, Faster Training: repeating smaller datasets speeds up learning via sampling biases. ICML 26.

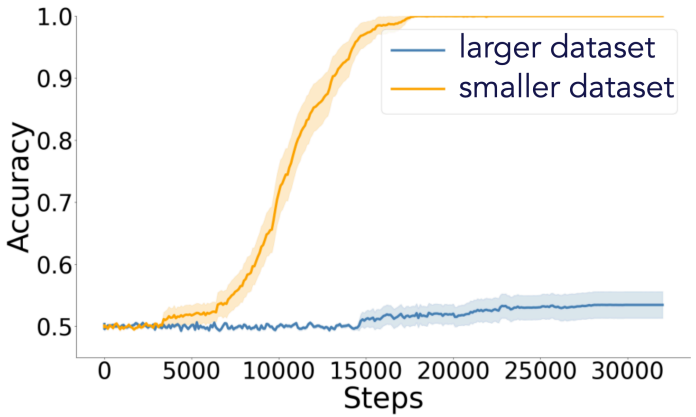
Effect of data repetition

*New regime: **limited data**, infinite compute*

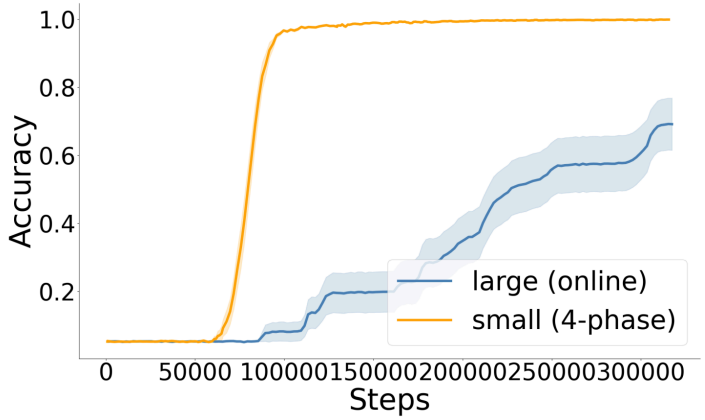
Coupled effect: improved data efficiency can lead to improved compute efficiency.

Data repetition as ***favorable optimization bias***.

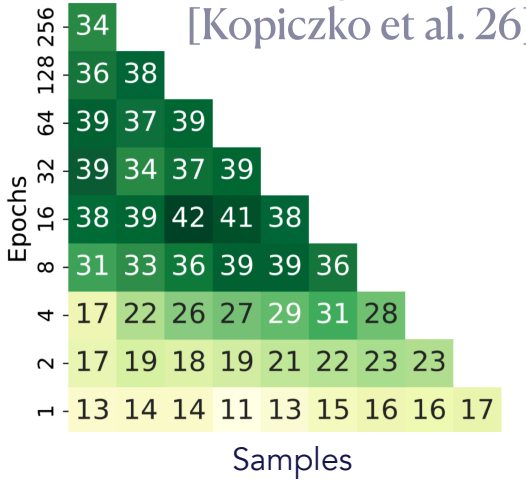
sparse parity



modular addition



LLM math reasoning [Kopiczko et al. 26]



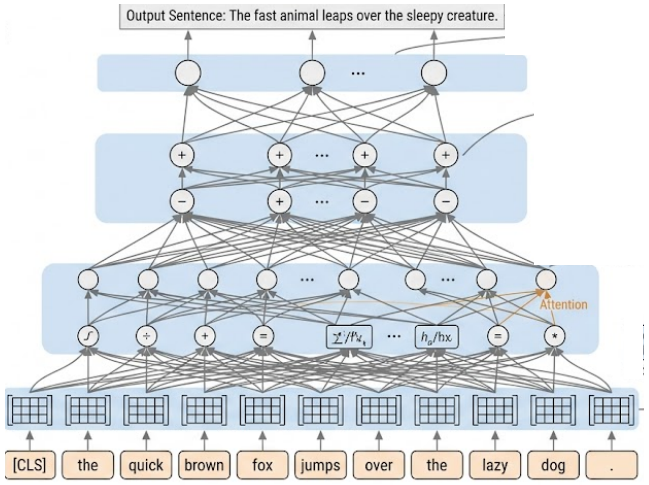
Kopiczko et al.: Data Repetition Beats Data Scaling in Long-CoT Supervised Fine-Tuning. 2026.

LEGL: Less Data, Faster Training: repeating smaller datasets speeds up learning via sampling biases. ICML 26.

Part 2: Efficient learning

Feature learning

in large models



improve LLMs

Sparse parity

in 2-layer nets

$$x \in \{\pm 1\}^d, y = \prod_{i \in S} x_i, \ |S| = k$$

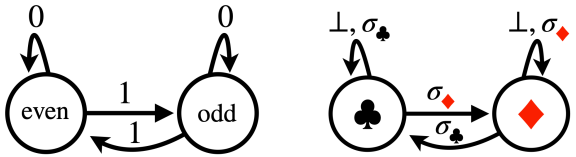
Change **data**-related factors

1. Richer supervision via **distillation**.
2. Data **repetition**

Efficient reasoning through sandboxes

Part 1: Compact reasoning **solutions**

automata



Parallel constructions + diagnosing models

Part 2: Improving **learning** efficiency

sparse parity

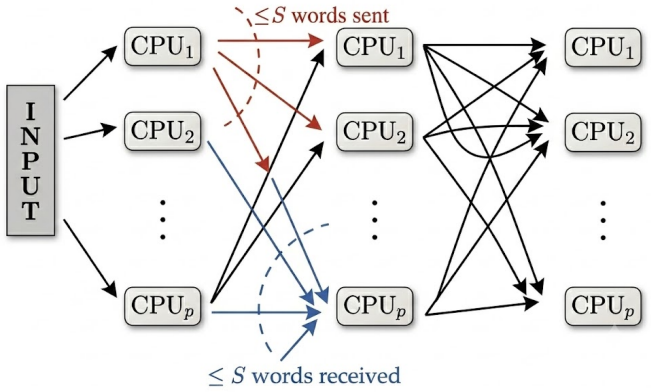
$$x = 1\ \underbrace{\text{-1 -1 1}}_{|S|=k}\ \text{-1 1 1 1 -1 1}$$

Data-related: distillation, repetition.

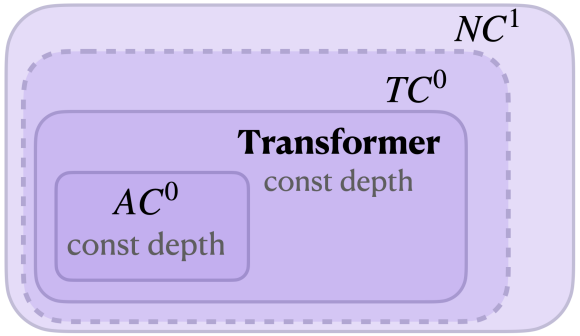
Choosing the sandbox is an art...

1. The choice determines the flavor of the results. *What is most suitable?*

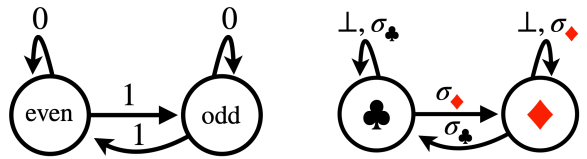
communication ↔ width



circuit ↔ depth



Krohn-Rhodes



Choosing the sandbox is an art...

1. The choice determines the flavor of the results. *What is most suitable?*

2. Abstractions are lossy, and the devil is often in the details. *Which ones matter?*

$$x_i^{(l)} = \phi\left(\sum_i \alpha_{i,j}^{(l-1)} x_j^{(l-1)}\right)$$

Residual connection? QK normalization? Numerical precision?

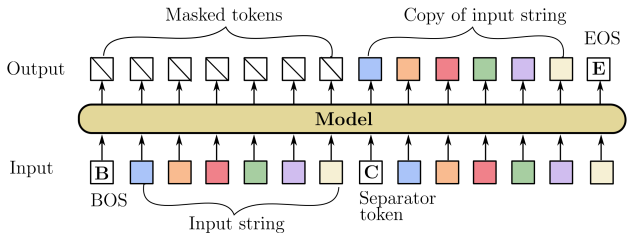
Impacting representation, learning, approximation.

Choosing the sandbox is an art...

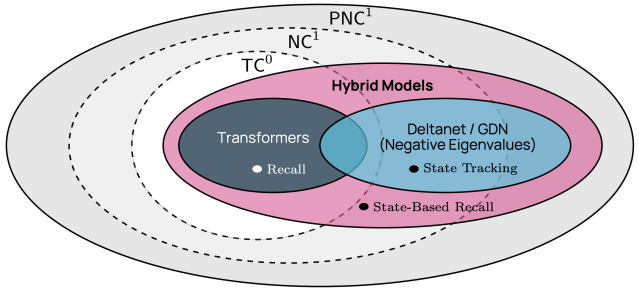
1. The choice determines the flavor of the results. *What is most suitable?*
2. Abstractions are lossy, and the devil is often in the details. *Which ones matter?*
3. Bitter lesson: **scalability** is crucial, often more than clever algorithms.

Bridging theory & practice

Architecture

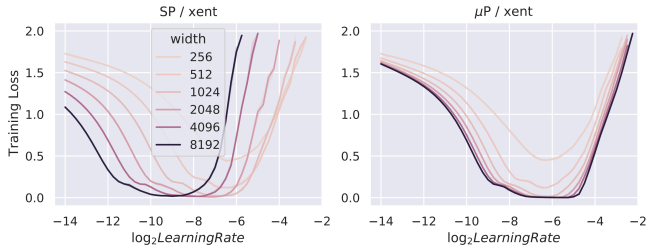


Limitations of RNN/SSM
[Jelassi et al. 24, Arora et al. 24]

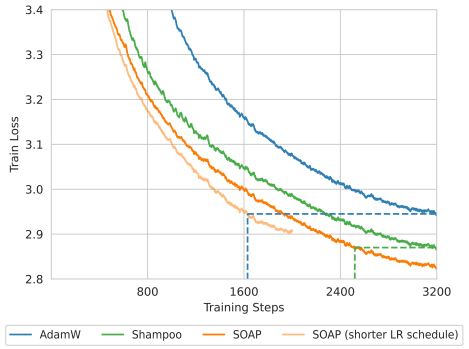


Hybrid models [Wen et al. 24, Merrill et al. 26]

Optimization

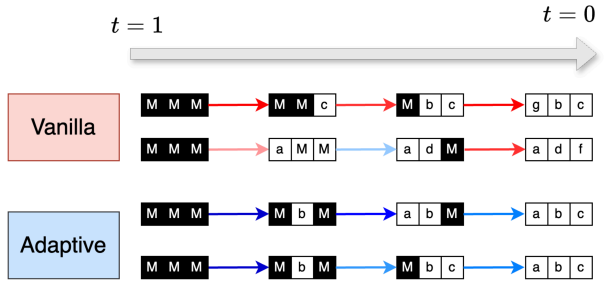


μP [Yang et al. 22]

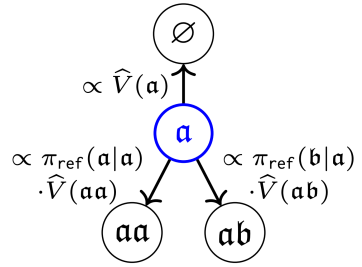


SOAP [Vyas et al. 24]

Inference



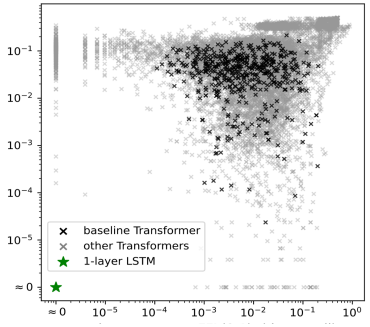
Masked prediction [Kim et al. 24]



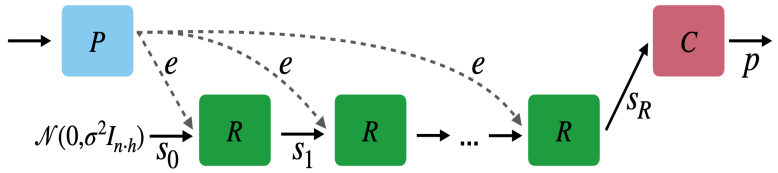
Test-time sampling [Rohatgi et al. 25]

Bridging theory & practice

Exact learning?

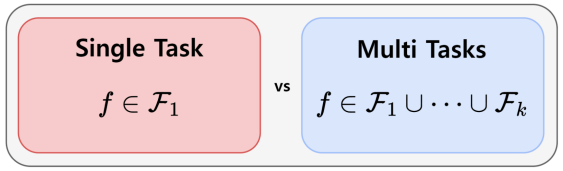


Practical role of **depth**?



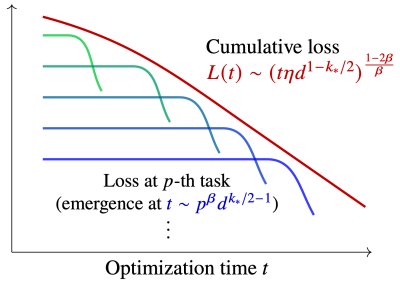
Recurrent depth [Geiping et al. 25]

Model for **data** (mixture)?



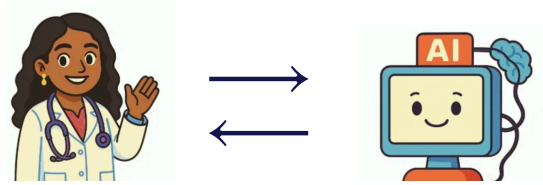
[Kim et al. 24]

Sources of **scaling** laws?



[Ren et al. 25]

AI-**human** collaboration?

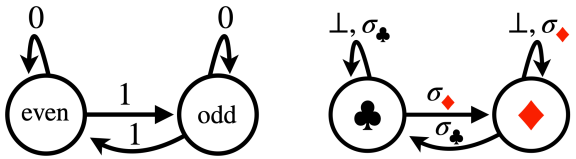


[Collina et al. 25]

Efficient reasoning through sandboxes

Part 1: Compact reasoning **solutions**

automata



Parallel constructions + diagnosing models

Part 2: Improving **learning** efficiency

sparse parity

$$x = 1\ \underbrace{-1\ -1\ 1}_{|S|=k}\ -1\ 1\ 1\ 1\ -1\ 1$$

Data-related: distillation, repetition.