

Foundations of Learning Reasoning Models

Nived Rajaraman (MSR), Nirmitt Joshi (TTIC)
and Nati Srebro (TTIC & Mirendil Inc)

COLT 2026



THE DARTMOUTH SUMMER RESEARCH PROJECT ON ARTIFICIAL INTELLIGENCE

Summer 1956

Proposed Topics:

1. Automatic Computers
2. How Can a Computer be Programmed to Use a Language
3. Neuron Nets
4. Theory of the Size of a Calculation
5. Self-improvement
6. Abstractions
7. Randomness and Creativity



Also: John Nash, Herbert Simon, Alan Newell, Arthur Samuel, and others

AI

Reasoning

- **Specify rules and knowledge**
- **Derive conclusions through multiple steps of applying the rules**

Learning

- Learn (end-to-end mapping) by fitting generic function approximator

Welcome to

```
EEEEEE LL      IIII ZZZZZZ  AAAA  
EE      LL      II     ZZ    AA   AA  
EEEEEE LL      II     ZZ    AAAAAA  
EE      LL      II     ZZ    AA   AA  
EEEEEE LLLLLL IIII ZZZZZZ  AA   AA
```

Eliza is a mock Rogerian psychotherapist

The original program was described by Joseph Weizenbaum in 1966

This implementation by Norbert Landsteiner 2005

ELIZA: How do you do. Please tell me your problem.

YOU: I forgot what I am supposed to teach my students.

EILZA: Do you think you are suppressing what you are supposed to teach your students?

YOU: I am not sure about that.

ELIZA: How long have you been not sure about that

YOU: For a few days.

ELIZA: What does that suggest to you ?

YOU:



Joseph
Weizenbaum
(1923-2008)

key: forget 5

decomp: * i forget *

reasmb: Why can't you remember (2) ?

reasmb: How often do you think of (2) ?

reasmb: Does it bother you to forget that ?

reasmb: Could it be a mental block ?

reasmb: Are you generally forgetful ?

reasmb: Do you think you are suppressing (2) ?

decomp: * did you forget *

reasmb: Why do you ask ?

reasmb: Are you sure you told me ?

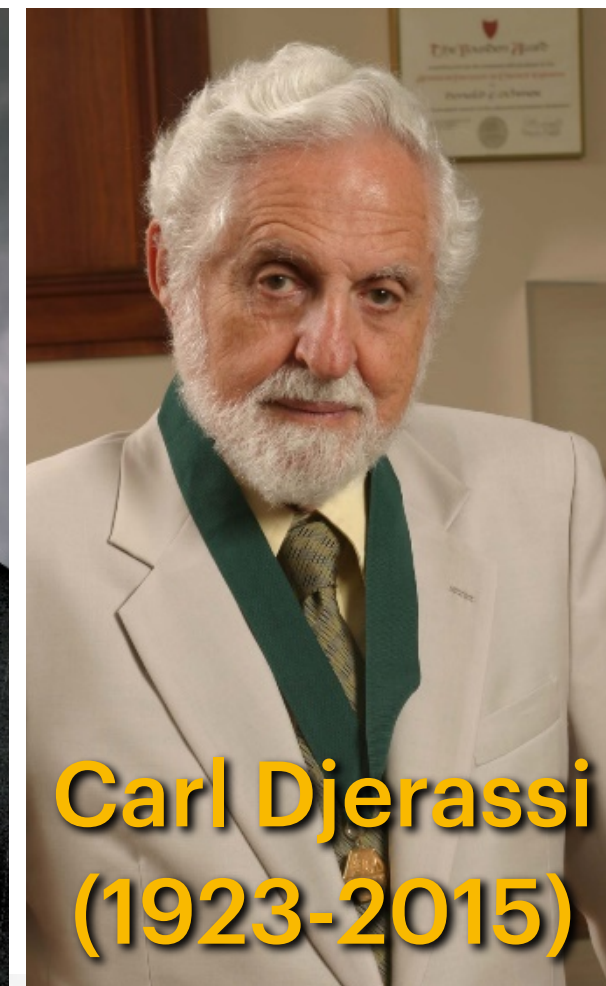
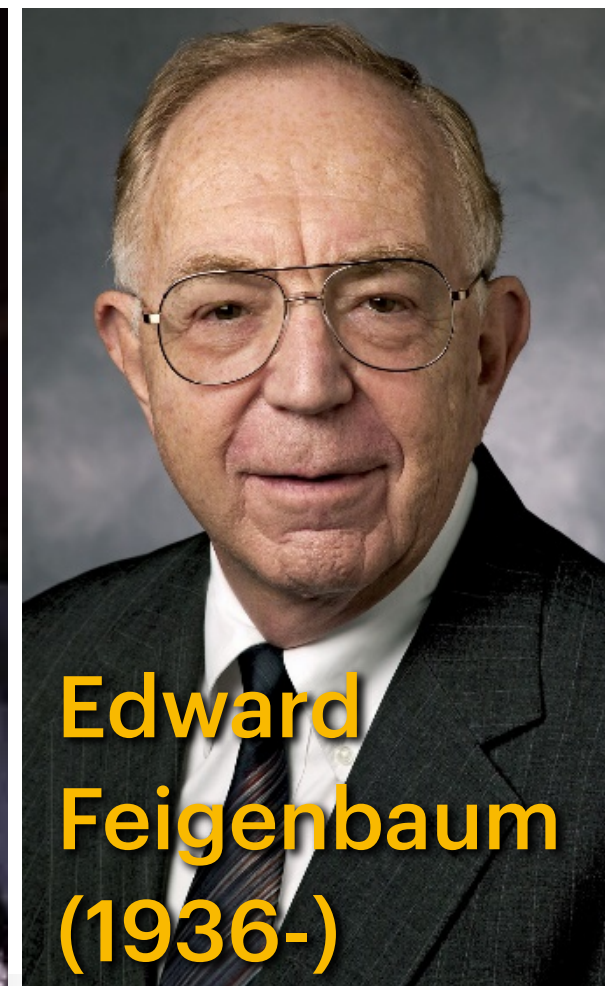
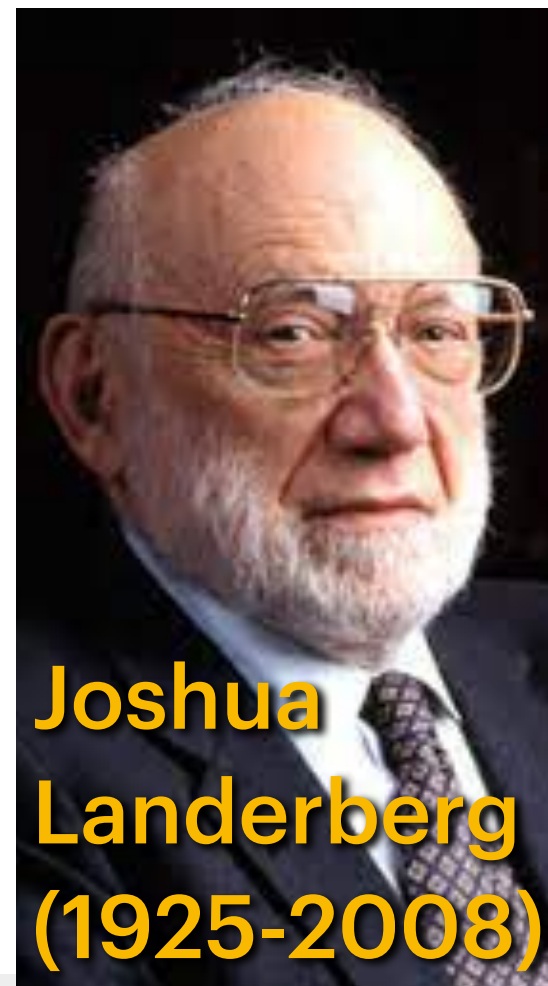
reasmb: Would it bother you if I forgot (2) ?

reasmb: Why should I recall (2) just now ?

reasmb: What do you suggest ?

Dendrel (1965-)

Identify
unknown
organic
compounds
based on
mass spec



```
WELCOME TO CONGEN, VERSION VI.  
#DEFINE MOLFORM C 12 H 14 O  
MOLECULAR FORMULA DEFINED
```

```
#DEFINE SUBSTRUCTURE Z
```

[Z is the structure required by constraints C2 and C3.]

```
(NEW SUBSTRUCTURE)
```

```
>RING 5
```

[C2: form five-membered ring.]

```
<LINK 1 1 1
```

[C2: add keto group.]

```
>ATNAME 6 O
```

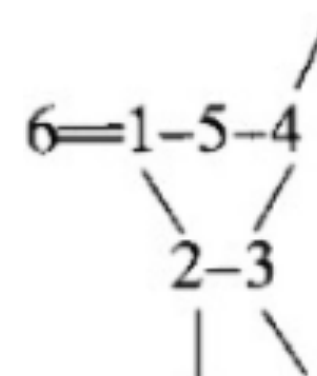
```
>HRANGE 2 1 1 3 1 2 4 1 2 5 2 2
```

[C3: atom 5 will have 2 H's, which will be alpha to the carbonyl;
atom 2 will have the third H alpha to C=O.]

```
>DRAW NUMBERED
```

```
SUBSTRUCTURE Z (HRANGES NOT INDICATED)
```

```
NON-C ATOMS: 6→O
```



```
#DEFINE SUBSTRUCTURE CH3
```

[C10: CH₃ is a methyl group.]

```
(NEW SUBSTRUCTURE)
```

```
>CHAIN
```

```
>HRANGE 1 3 3
```

```
>DONE
```

```
CH3 DEFINED
```

```
#DEFINE SUBSTRUCTURE V
```

[V is the vinyl group: C=C]

```
(NEW SUBSTRUCTURE)
```

```
RING 2
```



CYC is a Machine Reasoning AI that uses confided human common sense and knowledge (not patterns and statistics) for human-like cognitive processing

“Intelligence is ten million rules”, 1988
widely attributed but unverified



Douglas Lenat
(1950-2023)

```
(#$capitalCity #$France  
#$Paris)
```

```
(#$genls #$Tree-ThePlant  
#$Plant)
```

```
(#$implies  
  ($and  
    ($isa ?OBJ ?SUBSET)  
    ($genls ?SUBSET ?  
SUPERSET))  
  ($isa ?OBJ ?SUPERSET))
```

Started 1984

As of 2019 (latest whitepaper):

1.5M terms

25M rules

2000 PhD-person-years of work

AI

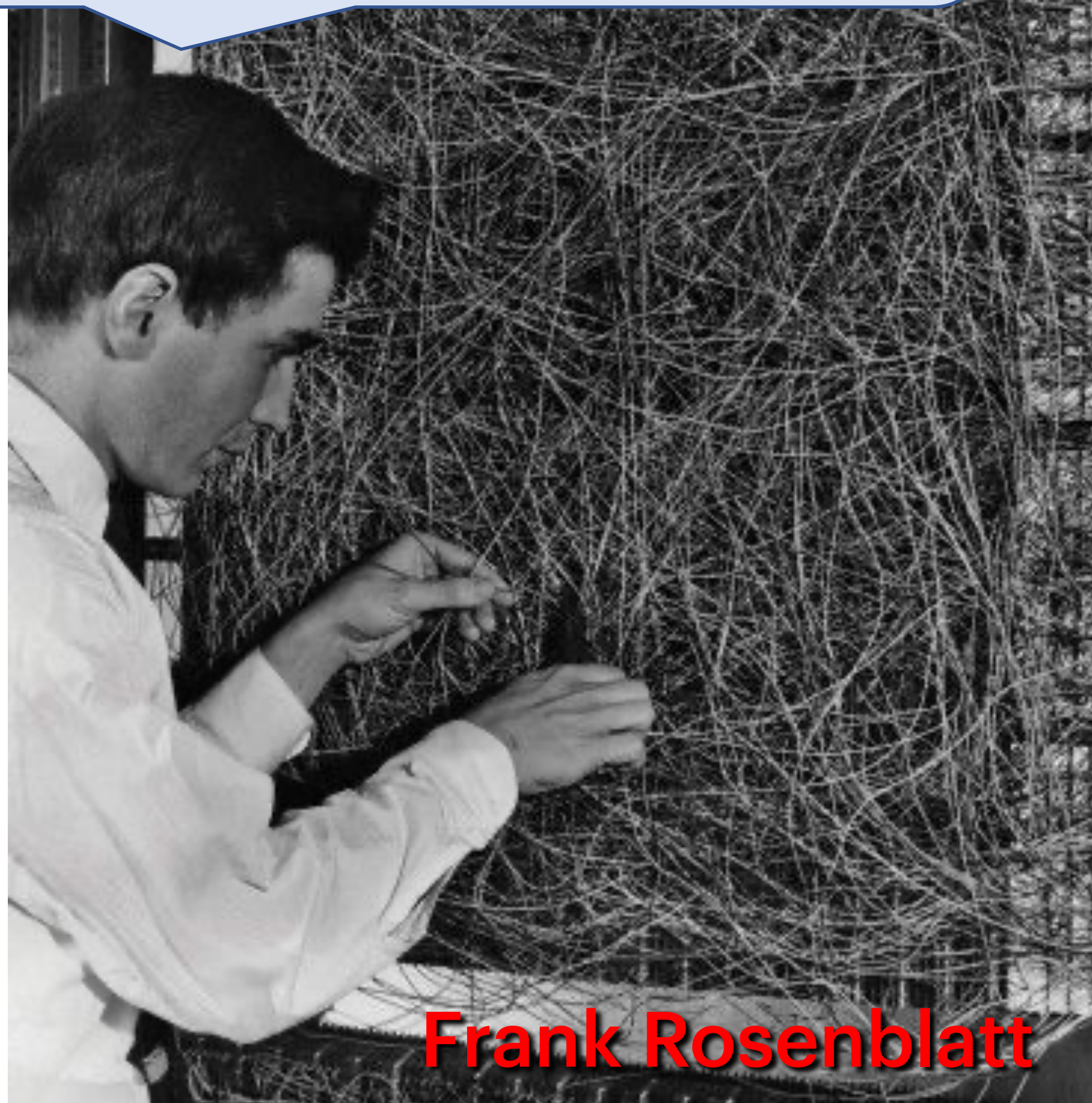
Reasoning

- Specify rules and knowledge
- Derive conclusions through multiple steps of applying the rules

Learning

- **Learn (end-to-end mapping) by fitting generic function approximator**

"the embryo of an electronic computer that [we] expect[s] will be able to walk, talk, see, write, reproduce itself and be conscious of its existence." (1958)



Frank Rosenblatt

Perceptron (1956): End-to-End Learning, no Reasoning

FIG. 1 — Organization of a biological brain. (Red areas indicate active cells, responding to the letter X.)

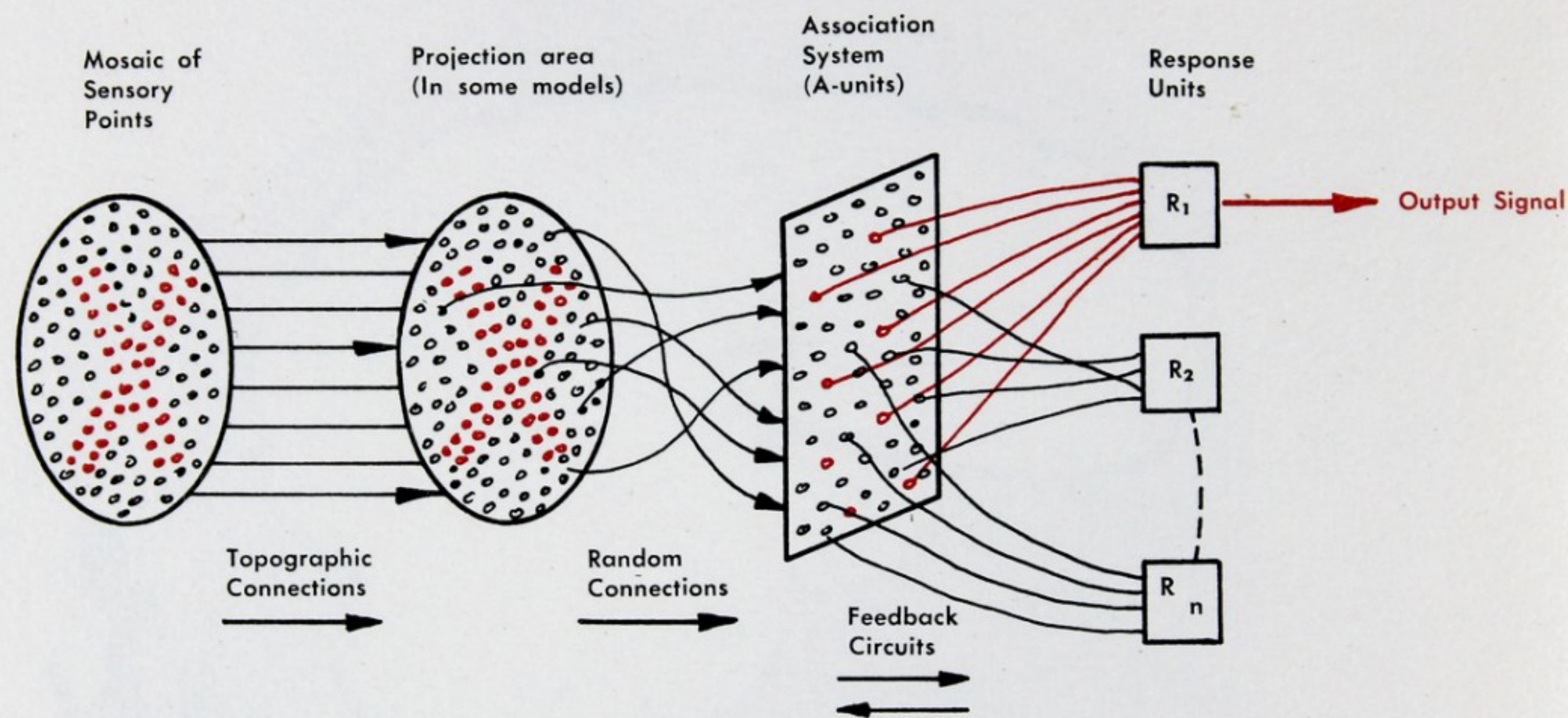


FIG. 2 — Organization of a perceptron.

1,000,000,000,000,000,000 less compute than Gemini Ultra

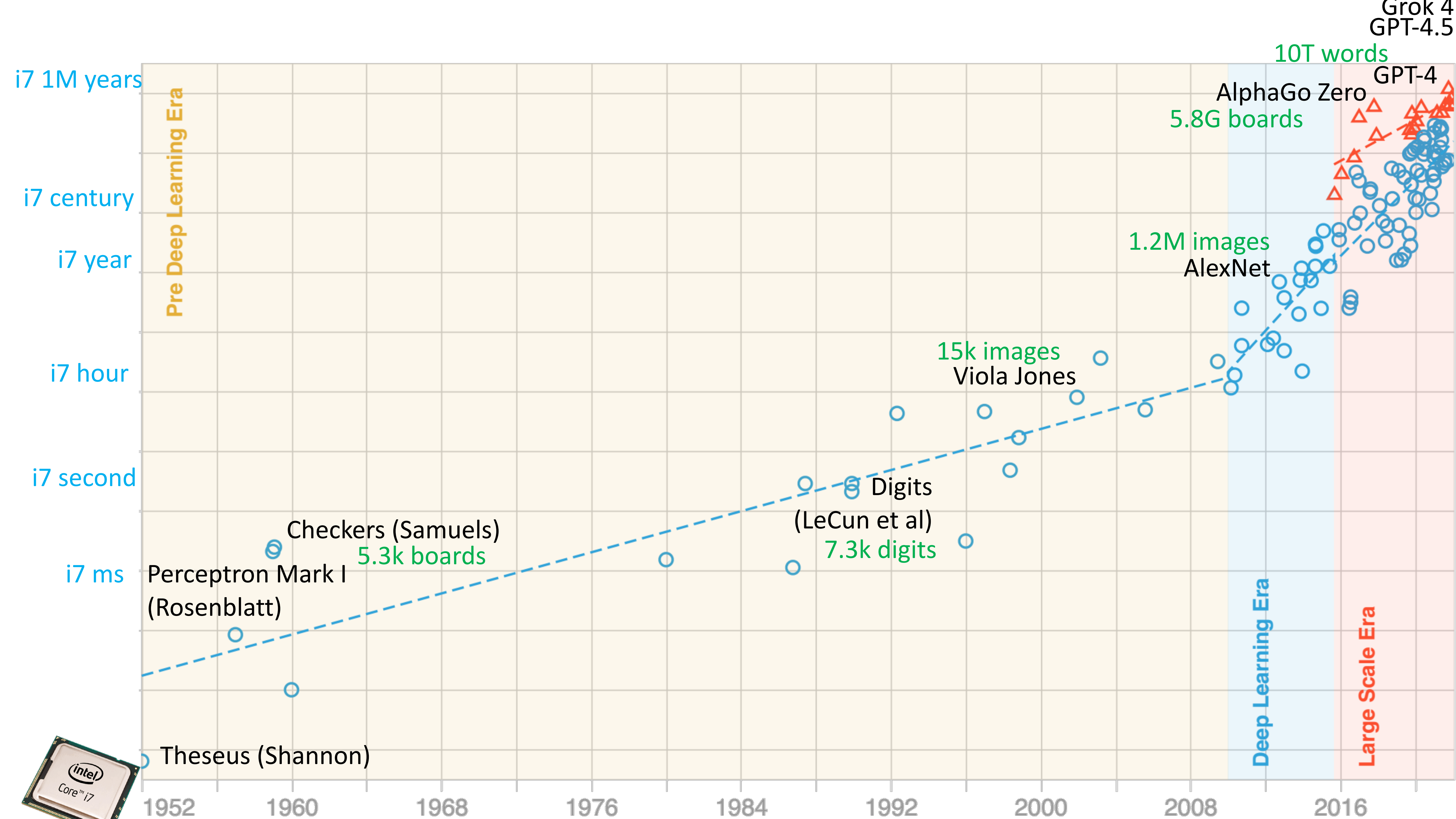
$\approx 10 \mu\text{s}$ vs 50 million years of i7 processing time (0.02 pico seconds vs 1,000 years of A100 GPUs)

Learning (Instead of Reasoning)

- 1956: Perceptron (in retrospect: SGD on hinge loss with random features approx kernel)
- 1956—1958: Solomonoff Inductive Inference (MDL/ERM on short programs)
- 1959: Arthur Samuel's Checkers ("neural net approach" will lead to "general purpose learning" but "long way before its practical", instead a specialized model which is "realizable at present time")
- 1984: Backpropagation (SGD on differentiable generic function approximator)

The Sweet Lesson:

1. Use a generic function approximator
(no experts, no linguists, no modeling)
2. Scale



i7 1M years
i7 century
i7 year
i7 hour
i7 second
i7 ms

Pre Deep Learning Era

Deep Learning Era

Large Scale Era

1952 1960 1968 1976 1984 1992 2000 2008 2016

Perceptron Mark I (Rosenblatt)
Checkers (Samuels)
5.3k boards

Digits (LeCun et al)
7.3k digits

15k images
Viola Jones

1.2M images
AlexNet

5.8G boards
AlphaGo Zero

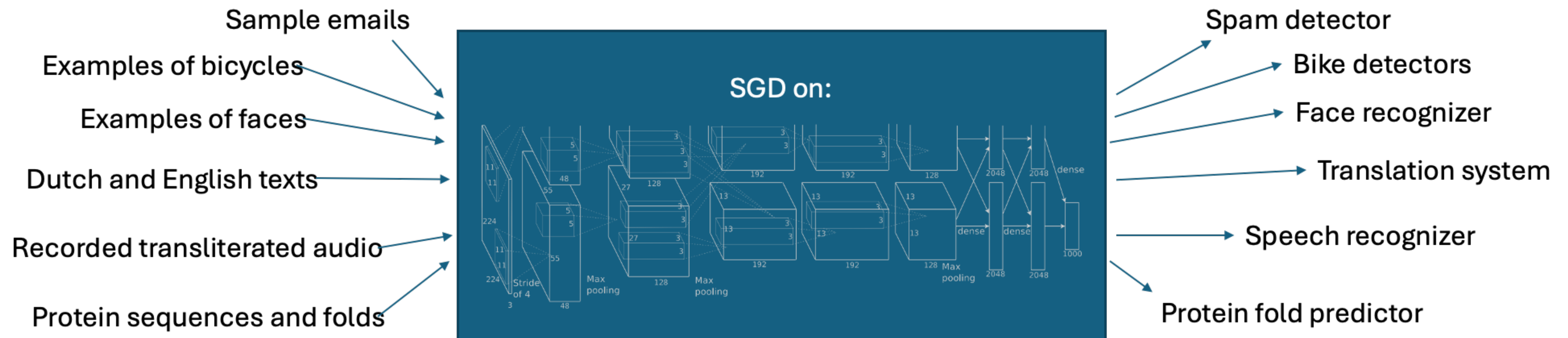
10T words

Grok 4
GPT-4.5

GPT-4

Learning (Instead of Reasoning)

- 1956: Perceptron (in retrospect: SGD on hinge loss with random features approx kernel)
- 1956—1958: Solomonoff Inductive Inference (MDL/ERM on short programs)
- 1959: Arthur Samuel's Checkers ("neural net approach" will lead to "general purpose learning" but "long way before its practical", instead a specialized model which is "realizable at present time")
- 1984: Backpropagation (SGD on differentiable generic function approximator)
- 1990s-2000s: Convex/Linear Models (SVMs, Boosting, etc; Viola-Jones face detector, kernels for vision, CRFs for translation, ad placement)
- 2012-: Non-convex differentiable generic function approximator (CNN, Transformers, etc)



2020:



How do I get from SAN to the Bahia?

Easiest is with Uber or Lyft

There were 32 pizzas ordered for lunch. 21 pizzas were eaten, but it turns out the restaurant sent 7 extra pizzas as a bonus. How many pizzas are left?

Write down only the answer.

26

Work step-by-step

To find out how many pizzas are left, we first add the number of pizzas that were ordered and the bonus pizzas, and then subtract the pizzas that were eaten.
Number of ordered pizzas: 32
Bonus pizzas: 7
Total pizzas: $32 + 7 = 39$
Pizzas eaten: 21
Number of pizzas left = Total pizzas - Pizzas eaten
Number of pizzas left = $39 - 21$
Number of pizzas left = 18
So, there are **18 pizzas left**.

Let $H(p)$ be the binary entropy, ie the entropy of a Bernoulli(p), and $KL(p||q)$ the KL-divergence between two Bernoullis. Define:
 $Q_\lambda(q) = \min_{(0 \leq p \leq 1/2)} \lambda KL(p||q) + H(p)$. We are interested in understanding $Q_\lambda(q)$ for $0 \leq q \leq 1/2$, and various values of λ . Let's start with $\lambda=1$.

To determine $Q_1(q)$ for $q \in [0, \frac{1}{2}]$, we need to minimize the function

$$F(p) = KL(p||q) + H(p)$$

over $p \in [0, \frac{1}{2}]$, where:

- $H(p) = -p \log p - (1-p) \log(1-p)$ is the binary entropy function,
- $KL(p||q) = p \log \frac{p}{q} + (1-p) \log \frac{1-p}{1-q}$ is the Kullback-Leibler divergence between two Bernoulli distributions.

Step 1: Simplify the Objective Function

First, expand and simplify $F(p)$:

$$\begin{aligned} F(p) &= KL(p||q) + H(p) \\ &= \left[p \log \frac{p}{q} + (1-p) \log \frac{1-p}{1-q} \right] - [p \log p + (1-p) \log(1-p)] \\ &= p \left(\log \frac{p}{q} - \log p \right) + (1-p) \left(\log \frac{1-p}{1-q} - \log(1-p) \right) \\ &= -p \log q - (1-p) \log(1-q). \end{aligned}$$

Step 2: Observe Linearity in p

The simplified function $F(p)$ is linear in p :

$$F(p) = -p \log q - (1-p) \log(1-q) = p(-\log q + \log(1-q)) - \log(1-q).$$

Since $-\log q + \log(1-q) \geq 0$ for $q \in [0, \frac{1}{2}]$, the coefficient of p is non-negative, and $F(p)$ is an increasing function of p over $[0, \frac{1}{2}]$.

Step 3: Find the Minimum

Because $F(p)$ is increasing in p , the minimum over $[0, \frac{1}{2}]$ occurs at $p = 0$:

$$Q_1(q) = F(0) = -\log(1-q).$$

Conclusion:

Therefore, for $\lambda = 1$ and $q \in [0, \frac{1}{2}]$, we have:

$$Q_1(q) = -\log(1-q).$$

Answer:

An explicit formula: $Q_1(q) = -\log(1-q)$ for q between 0 and $\frac{1}{2}$; that is,

$$Q_1(q) = -\log(1-q)$$

AI

Reasoning

- Specify rules and knowledge
- Derive conclusions through multiple steps of applying the rules

Learning

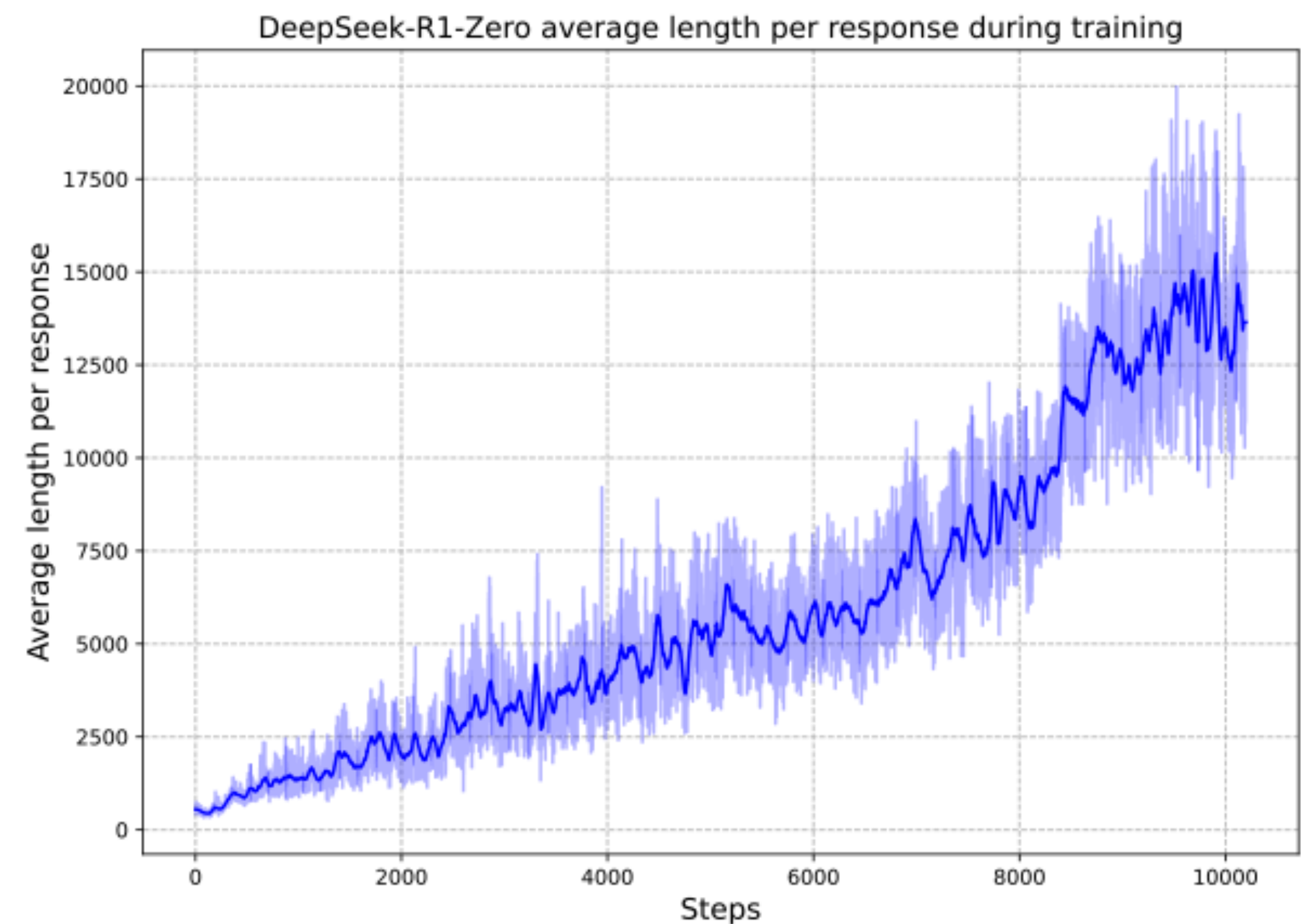
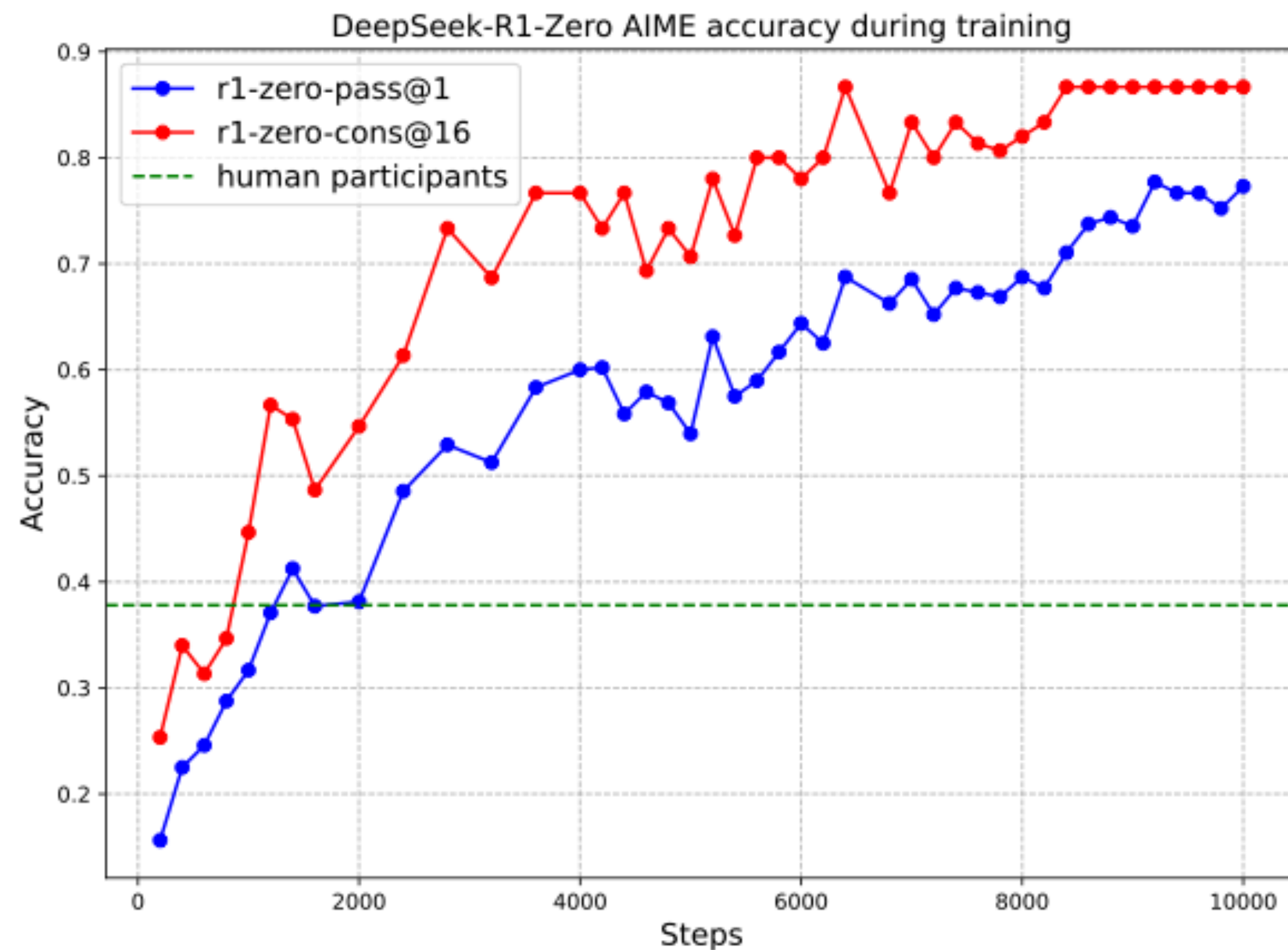
- Learn (end-to-end mapping) by fitting generic function approximator

Learning to Reason

- **Learn to step**
 - **Perform many steps**
- 

Training models to think

DeepSeek-R1: model reasoning encapsulated within <think> </think> tags, final answer within <answer> </answer>; model trained via SFT + RL

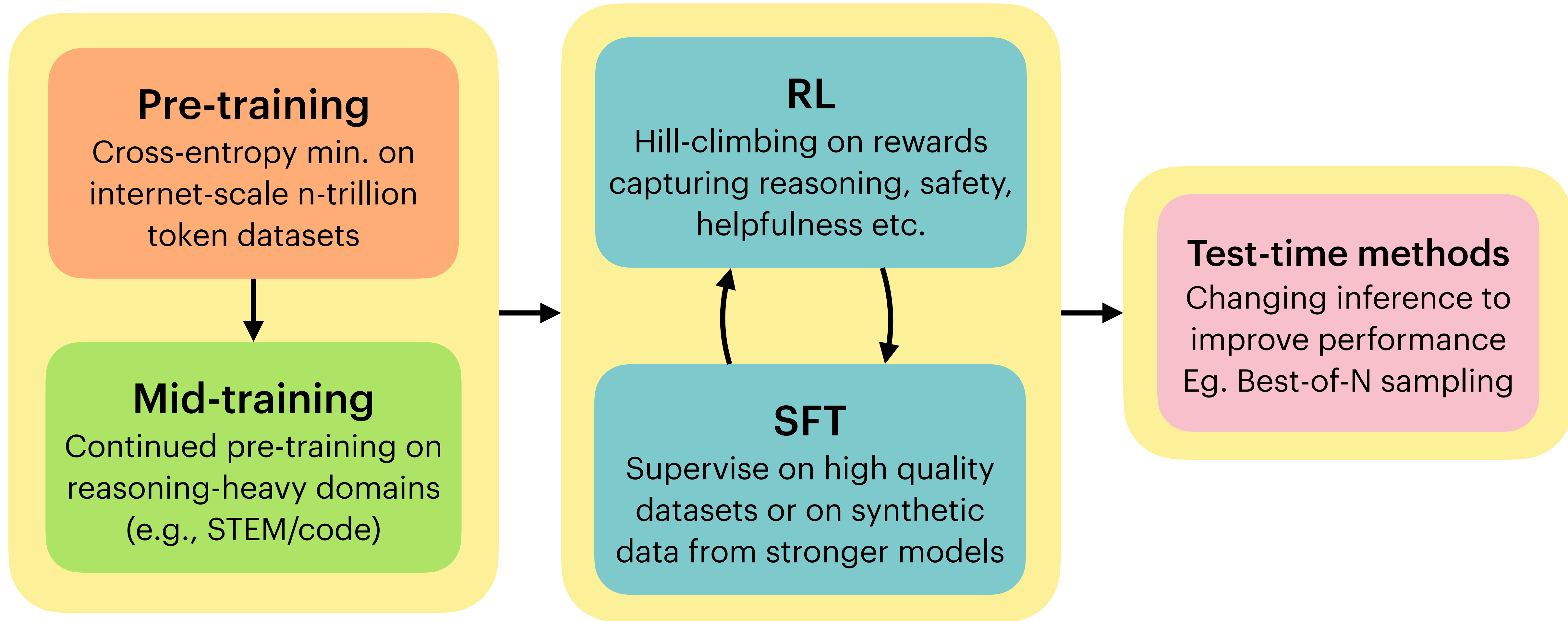


Long reasoning / test-time search can emerge through RL **under the right conditions.**

Reasoning/Computing in Deep Models vs Autoregressive CoT

- Deep Models: layers are compute/reasoning steps
 - #params (hence #samples) grows with #steps
 - Hard to train beyond depth ~ 100 (too much latent)
- Recurrent/Looped Models:
 - #params independent of #steps
 - size, hence #params and #samples still grows with memory (state size)
 - Hard to train beyond ~ 100 steps (too much latent)
- **Autoregressive CoT**
 - Attention allows decoupling #params from #steps, memory
 - Direct observation on reasoning steps (not always latent!)
 - **“Think” in same language as the observed data**

Canonical LLM training process:



Length of reasoning traces grow

Learning to Reason with Autoregressive CoT

Data:

- End-to-End (question-answer, or question/“environment”-reward/verifier)
 - Other/“Practice”/Curriculum end-to-end
 - Explicit full thinking traces
 - Other (not explicitly thinking)
- } **“Positive Distribution Shift”**
(training on other data instead of target task improves learning)
⇒ **Data Set Selection**

Compute/Algorithms:

- Training time: How can we train with end-to-end supervision?
How can we utilize combination of different data sources?
- Test time: How can we use the learned model for reasoning?

Outline

Part I: *A theoretical model for learning with CoTs*

- *Statistical and Computational Advantage*

Part II: *Learning from Heterogeneous Data in Pretraining*

- *Heterogeneity, Coverage, and Stochasticity*

Part III: *RL post-training and Beyond*

- *RL from Coverage, Verifiers in Inference Time Scaling, and Learning Verifiers*

Part IV: *Future Directions*

Part I. Foundations of Learning from With Chain-of-Thought

Statistical and Computational Advantages of CoT

Time Invariant Autoregressive Models

“What is the GCD of 8 and 12?”

Factors of 12: 1, 2, 3, 4, 6, 12, and of 8: 1, 2, 4, 8. So the GCD is 4.

Time Invariant Autoregressive Models

Prompt: \mathbf{x}

“What is the GCD of 8 and 12?”

Factors of 12: 1, 2, ...

4.

(Last token of CoT)
Answer: $y = f^{e2e-T}(\mathbf{x})$

Next-token generator $f: \Sigma^* \rightarrow \Sigma$

$\bar{f}: s \mapsto \text{append}(s, f(s)) \in \Sigma^{|s|+1}$

$$f^{\text{CoT}-T}(\mathbf{x}) = \underbrace{\bar{f} \circ \bar{f} \circ \dots \circ \bar{f} \circ \bar{f}}_{T \text{ times}}(\mathbf{x})$$

Consider any base class \mathcal{F} .

End-to-end class:

$$\mathcal{F}^{e2e-T} = \{f^{e2e-T} : \Sigma^* \rightarrow \Sigma \mid f \in \mathcal{F}\}$$

CoT vs E2E Supervision

(Unknown) Input distribution \mathcal{D} and $f_\star \in \mathcal{F}$.

$$\mathbf{x}_1, \dots, \mathbf{x}_m \sim_{iid} \mathcal{D}$$

E2E supervision

("What is the GCD of 8 and 12?", "4")

$$S_{e2e} = \left((\mathbf{x}_i, y_i) : i \in [m], y_i = f_\star^{e2e-T}(\mathbf{x}_i) \right)$$

CoT supervision

("What is the GCD of 8 and 12?", "Factors of 12: 1,2,3,4,6,12 and of 8: 1,2,4,8. So GCD is 4")

$$S_{CoT} = \left((\mathbf{z}_i := f_\star^{CoT-T}(\mathbf{x}_i)) : i \in [m] \right)$$

Learning Objective: Maximize the final answers accuracy (i.e., the end-to-end disagreement loss),

$$L_{\mathcal{D}, f_\star}^{01}(\hat{f}) := \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \left[\mathbf{1} \left\{ \hat{f}(\mathbf{x}) \neq f_\star^{e2e-T}(\mathbf{x}) \right\} \right]$$

What is the statistical and computational advantage of CoT supervision?

CoT datasets are the currency of Supervised Fine Tuning (SFT)

CoT datasets are usually collected from larger models, or filtering self-generated data
Expensive to collect, but a vital part of training reasoning models

[home](#) [blog](#) [github](#) [huggingface](#)



A [DataComp](#) and [Bespoke Labs](#) community effort to curate the best open reasoning datasets.

Our latest release is [OpenThoughts3](#), the new SOTA open reasoning data recipe. Read the [blog](#) and the [full paper](#) for more detail!

ThoughtSource ⚡

A framework for the science of machine thinking

[Datasets](#) • [Tutorial notebook](#) • [Installation guide](#) • [Dataset Annotator](#)

ThoughtSource is a central, open resource and community centered on data and tools for chain-of-thought reasoning in large language models ([Wei 2022](#)). Our long-term goal is to enable trustworthy and robust reasoning in advanced AI systems for driving scientific research and medical practice.

Input:

Evaluate $-7*x**2 + 7*x + 5$ at $x = 1$

Target:

<scratch>

$-7*x**2$: -7

$7*x$: 7

5: 5

</scratch>

total: 5

A carnival snack booth made \$50 selling popcorn each day. It made three times as much selling cotton candy. For a 5-day activity, the booth has to pay \$30 rent and \$75 for the cost of the ingredients. How much did the booth earn for 5 days after paying the rent and the cost of ingredients?

How much did the booth make selling cotton candy each day? ** The booth made $\$50 \times 3 = \150 selling cotton candy each day.

How much did the booth make in a day? ** In a day, the booth made a total of $\$150 + \$50 = \$200$.

How much did the booth make in 5 days? ** In 5 days, they made a total of $\$200 \times 5 = \1000 .

How much did the booth have to pay? ** The booth has to pay a total of $\$30 + \$75 = \$105$.

How much did the booth earn after paying the rent and the cost of ingredients? ** Thus, the booth earned $\$1000 - \$105 = \$895$.

CoT vs. E2E: Statistical Rates for Finite and Infinite Classes

- CoT feedback provides more information than just E2E feedback
- Does this manifest in a difference in sample complexity?

Finite Classes: Statistical Sample Complexity

Base class Measure	End-to-End Samples	CoT-Samples
	$m^{e^{2\epsilon}}$	m^{CoT}
$ \mathcal{F} < \infty$	$\lesssim \frac{\log \mathcal{F} }{\epsilon}$	$\lesssim \frac{\log \mathcal{F} }{\epsilon}$

E2E-Consistent / ERM under $\mathcal{F}^{e^{2\epsilon-T}}$

Return $\hat{f}^{e^{2\epsilon-T}}$ for $\hat{f} \in \mathcal{F}$ s.t. $\forall i \in [m], \hat{f}^{e^{2\epsilon-T}}(\mathbf{x}_i) = y_i$

- Because of **time invariance** (e.g., parameter sharing): $|\mathcal{F}^{e^{2\epsilon-T}}|, |\mathcal{F}^{\text{CoT-T}}| \leq |\mathcal{F}|$
- Time invariance allows sample complexity independent of T .

E2E feedback: Analyzing Sample Complexity

Bounding $\text{VCdim}(\mathcal{F}^{e^{2e-T}})$. Suppose vocabulary $\Sigma = \{0,1\}$

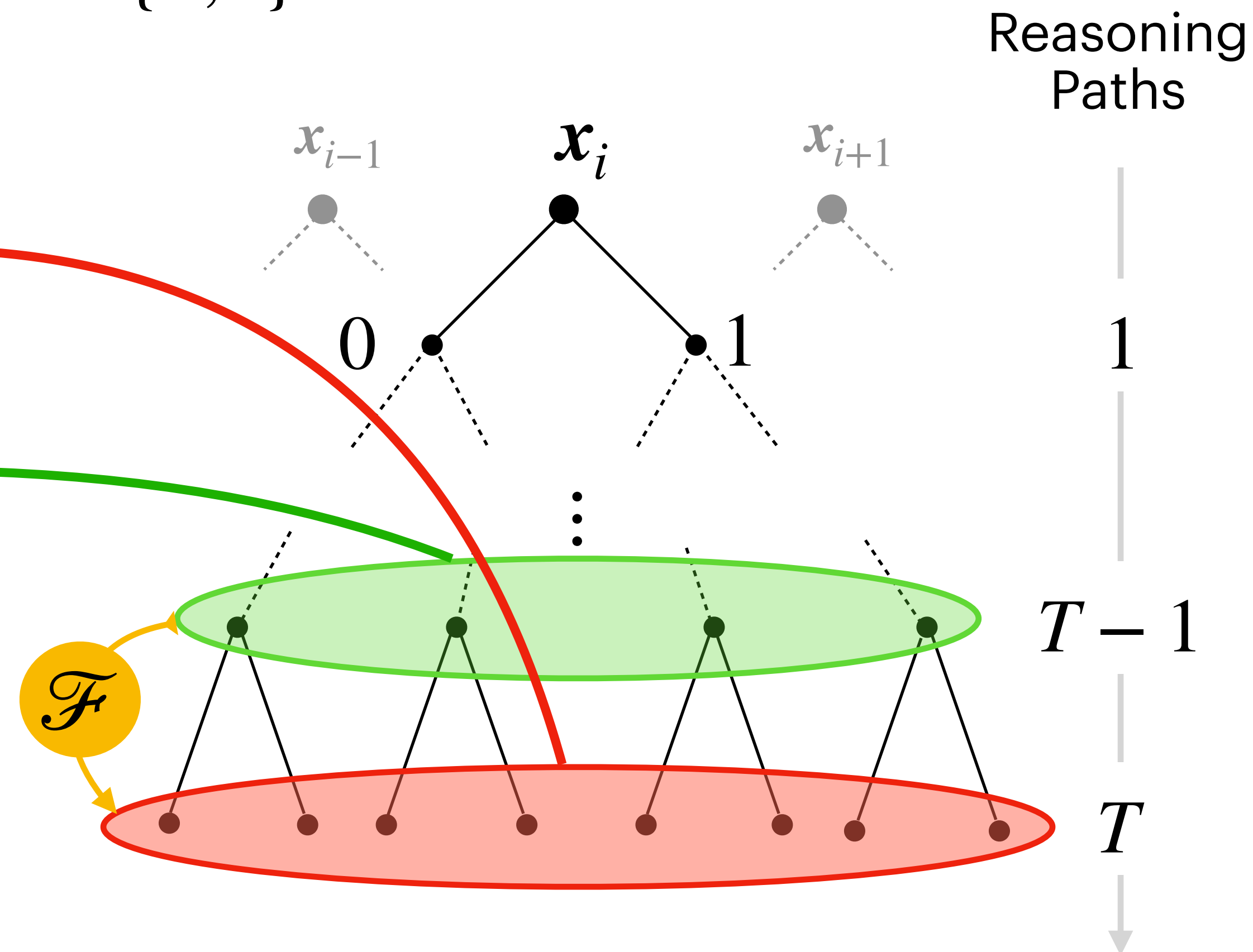
- Suffices to bound the growth function $\Gamma_{\mathcal{F}^{e^{2e-T}}}(m)$ of $\mathcal{F}^{e^{2e-T}}$ on m prompts $\{\mathbf{x}_i\}_{i=1}^m$.

- $\mathcal{F}^{e^{2e-T}}$ can express $m2^{T-1}$ behaviors at $T - 1$.

$$\begin{aligned} \implies \Gamma_{\mathcal{F}^{e^{2e-T}}}(m) &\leq \Gamma_{\mathcal{F}}(m2^{T-1}) \\ &\leq \left(\frac{em2^{T-1}}{\text{VCdim}(\mathcal{F})} \right)^{\text{VCdim}(\mathcal{F})} \end{aligned}$$

$$\implies \text{VCdim}(\mathcal{F}^{e^{2e-T}}) \lesssim T \cdot \text{VCdim}(\mathcal{F})$$

This is tight in the worst-case.



Summarizing,

Base class	End-to-End samples m^{e2e}	CoT samples m^{CoT}
$ \mathcal{F} < \infty$	$\gtrsim \frac{\log \mathcal{F} }{\epsilon}$	$\gtrsim \frac{\log \mathcal{F} }{\epsilon}$
$\text{VCdim}(\mathcal{F}) < \infty$	$\gtrsim \frac{T \cdot \text{VCdim}(\mathcal{F})}{\epsilon}$ $\gtrsim T \cdot \text{VCdim}(\mathcal{F})$???

CoT feedback: Sample Complexity for VC classes

• Given

$S = ((\mathbf{x}_1, \mathbf{z}_1), \dots, (\mathbf{x}_m, \mathbf{z}_m))$ where $\mathbf{z}_i = f_{\star}^{\text{CoT-}T}(\mathbf{x}_i)$,

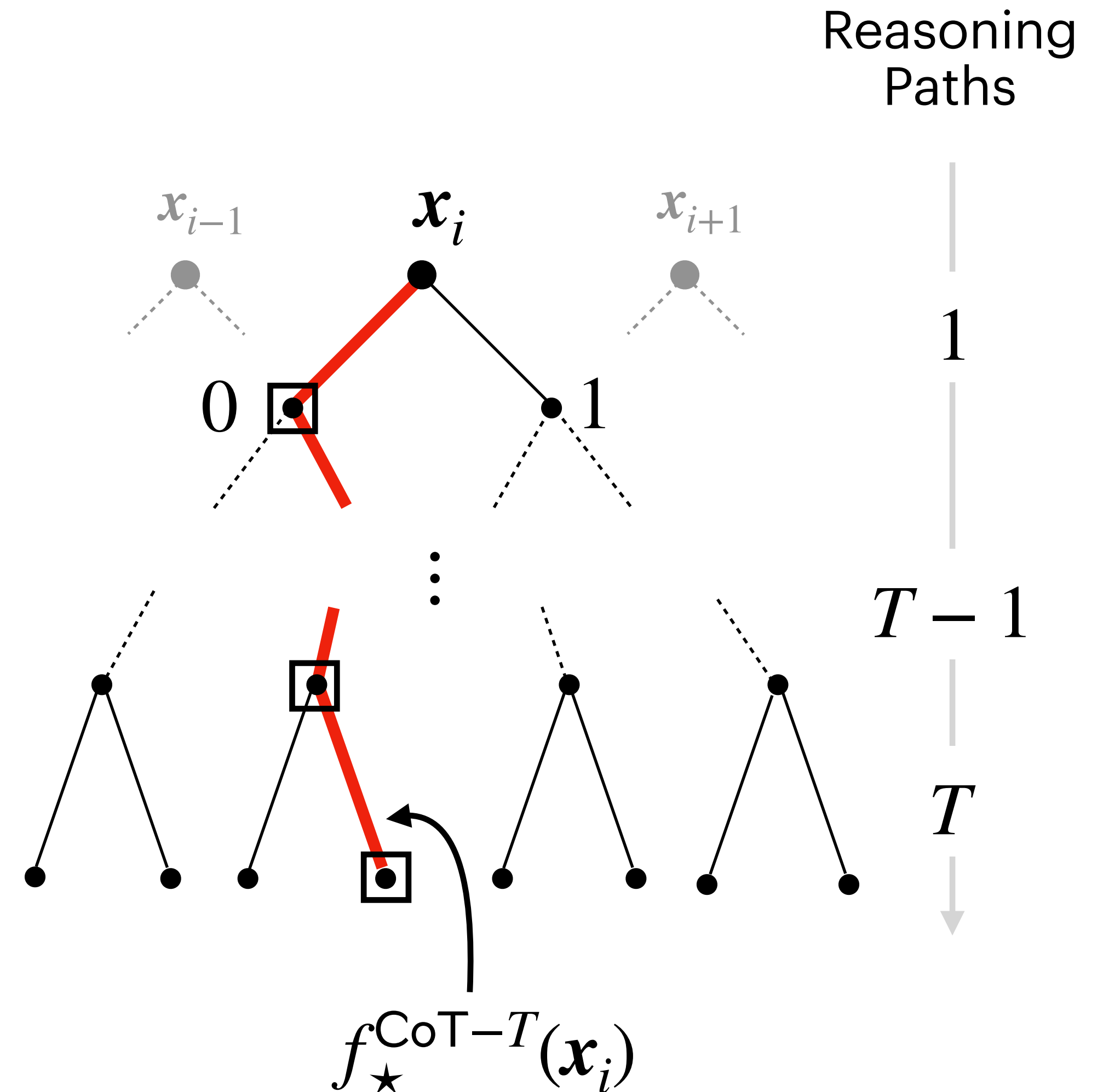
CoT-Consistent / ERM under $\mathcal{F}^{\text{CoT-}T}$

Return $\hat{f}^{e^{2e^{-T}}}$ for $\hat{f} \in \mathcal{F}$ s.t. $\hat{f}^{\text{CoT-}T}(\mathbf{x}_i) = \mathbf{z}_i$.

CoT-loss, $\ell_f(\mathbf{x}) = \mathbf{1}\{f^{\text{CoT-}T}(\mathbf{x}_i) \neq f_{\star}^{\text{CoT-}T}(\mathbf{x}_i)\}$ determined by the “behavior” of f on prefixes indicated by square boxes (Any deviation from red path makes the CoT-loss = 1)

Define, CoT-loss class, $\mathcal{L}^{\text{CoT-}T} = \{\ell_f : f \in \mathcal{F}\}$

Then, $\Gamma_{\mathcal{L}^{\text{CoT-}T}}(m) \leq \Gamma_{\mathcal{F}}\left(m, T \left(\frac{emT}{\text{VCdim}(\mathcal{F})}\right)^{\text{VCdim}(\mathcal{F})}\right) \implies \text{VCdim}(\mathcal{L}^{\text{CoT-}T}) \leq \text{VCdim}(\mathcal{F}) \cdot \log(T)$



CoT feedback: Statistical Benefits

Base class Measure	End-to-End Learnability m^{e2e}	CoT-Learnability m^{CoT}
$ \mathcal{F} < \infty$	$\lesssim \frac{\log \mathcal{F} }{\varepsilon}$	$\lesssim \frac{\log \mathcal{F} }{\varepsilon}$
$\text{VCdim}(\mathcal{F}) < \infty$	$\lesssim T \cdot \text{VCdim}(\mathcal{F})$ $\lesssim T \cdot \frac{\text{VCdim}(\mathcal{F})}{\varepsilon}$	$\lesssim \text{VCdim}(\mathcal{F}) \cdot \frac{\log T}{\varepsilon}$

Summary of Sample Complexity Bounds

Base class Measure	End-to-End Learnability m^{e2e}	CoT-Learnability m^{CoT}
Learning Rule	e2e-Consistent	CoT-Consistent
$ \mathcal{F} < \infty$	$\lesssim \frac{\log \mathcal{F} }{\epsilon}$	$\lesssim \frac{\log \mathcal{F} }{\epsilon}$
$\text{VCdim}(\mathcal{F}) < \infty$	$\lesssim \text{VCdim}(\mathcal{F}) \cdot \frac{T}{\epsilon}$ $\gtrsim T \cdot \text{VCdim}(\mathcal{F})$	$\lesssim \text{VCdim}(\mathcal{F}) \cdot \frac{\log T}{\epsilon}$
Improved Bounds	$\lesssim \text{ATdim}(\mathcal{F}) \cdot \frac{\log T}{\epsilon}$ [Hanneke et al. 26]	$\lesssim \text{VCdim}(\mathcal{F}) \cdot \frac{\log T}{\mathcal{F}(\epsilon)}$ [Altabaa et al. 25]

$$\begin{aligned} \text{VCdim}(\mathcal{F}) &\leq \text{ATdim}(\mathcal{F}) \\ &\leq \text{Ldim}(\mathcal{F}) \end{aligned}$$

Sample Complexity of Autoregressive Reasoning:
Chain-of-Thought vs. End-to-End

Steve Hanneke¹, Idan Mehal², and Shay Moran³

CoT Information: Improved Sample Complexity under Chaos
[Hanneke et al. 26]

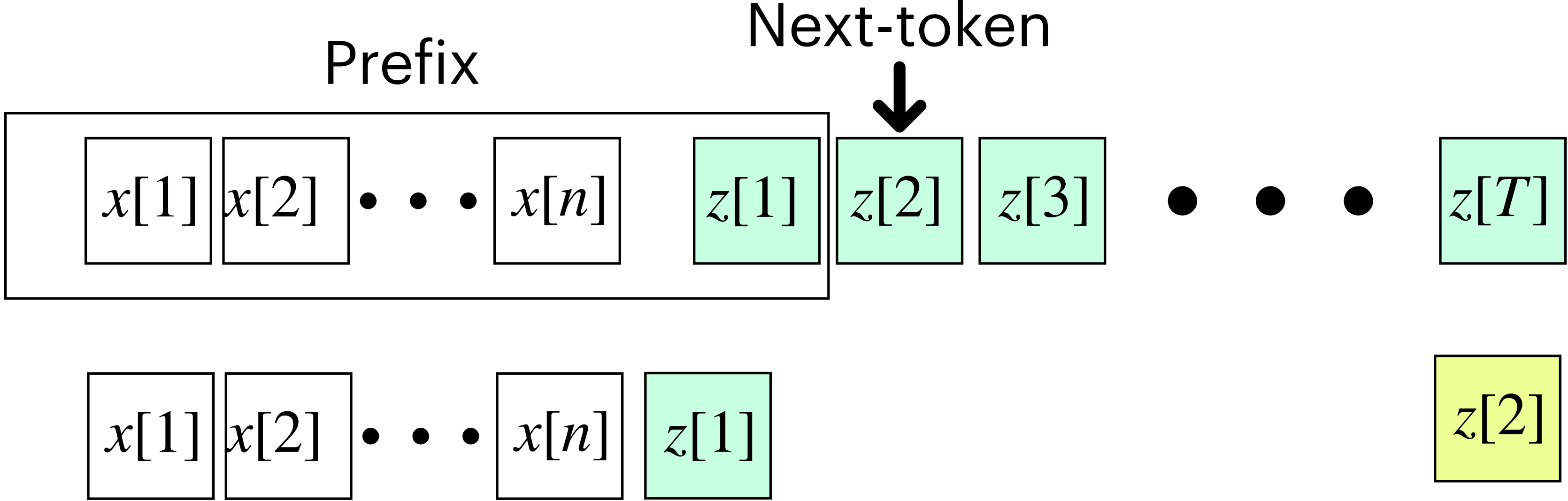
Awni Altabaa¹ Omar Montasser² John Lafferty³

CoT vs. E2E: Computational Complexity

- How does the step-by-step supervision provided by CoT feedback help computationally?
- Lower bounds in the absence of CoT feedback?

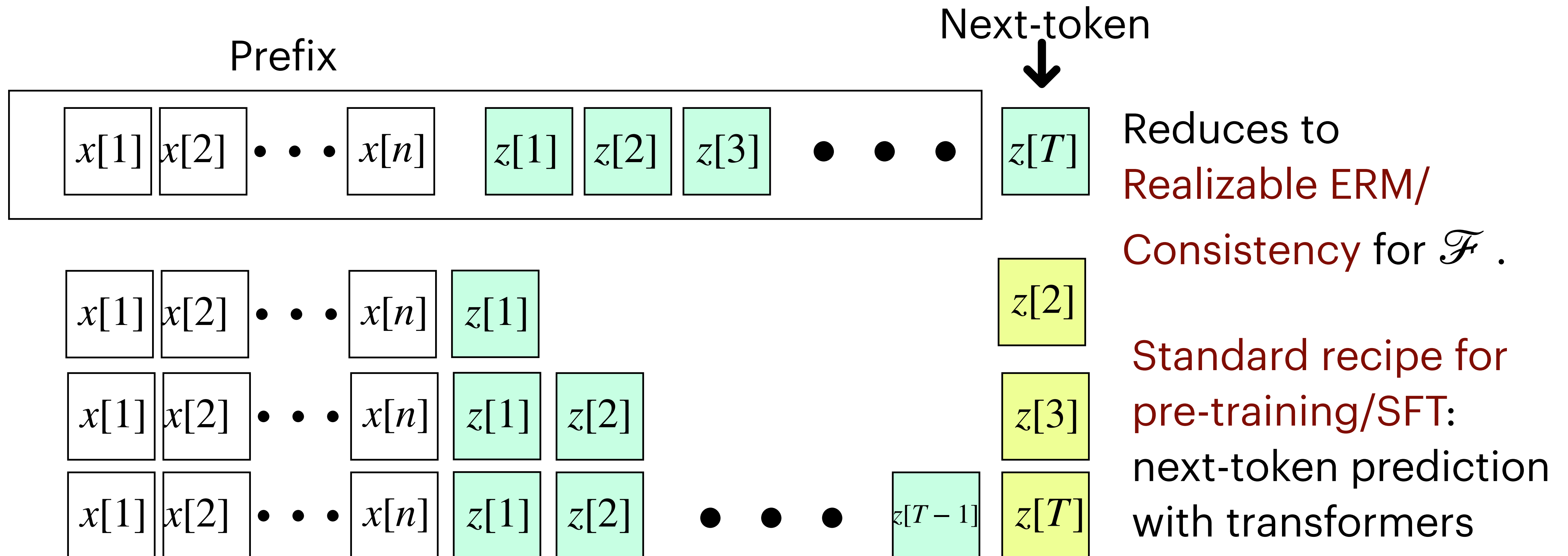
CoT feedback: Computational Advantage

	End-to-End Supervision	CoT Supervision
Computational Complexity		Easy! Whenever the base class has a realizable ERM (consistency) oracle.



CoT feedback: Computational Advantage

	End-to-End Supervision	CoT Supervision
Computational Complexity	Hard. Not necessarily learnable computationally efficiently.	Easy! Whenever the base class has a realizable ERM (consistency) oracle.

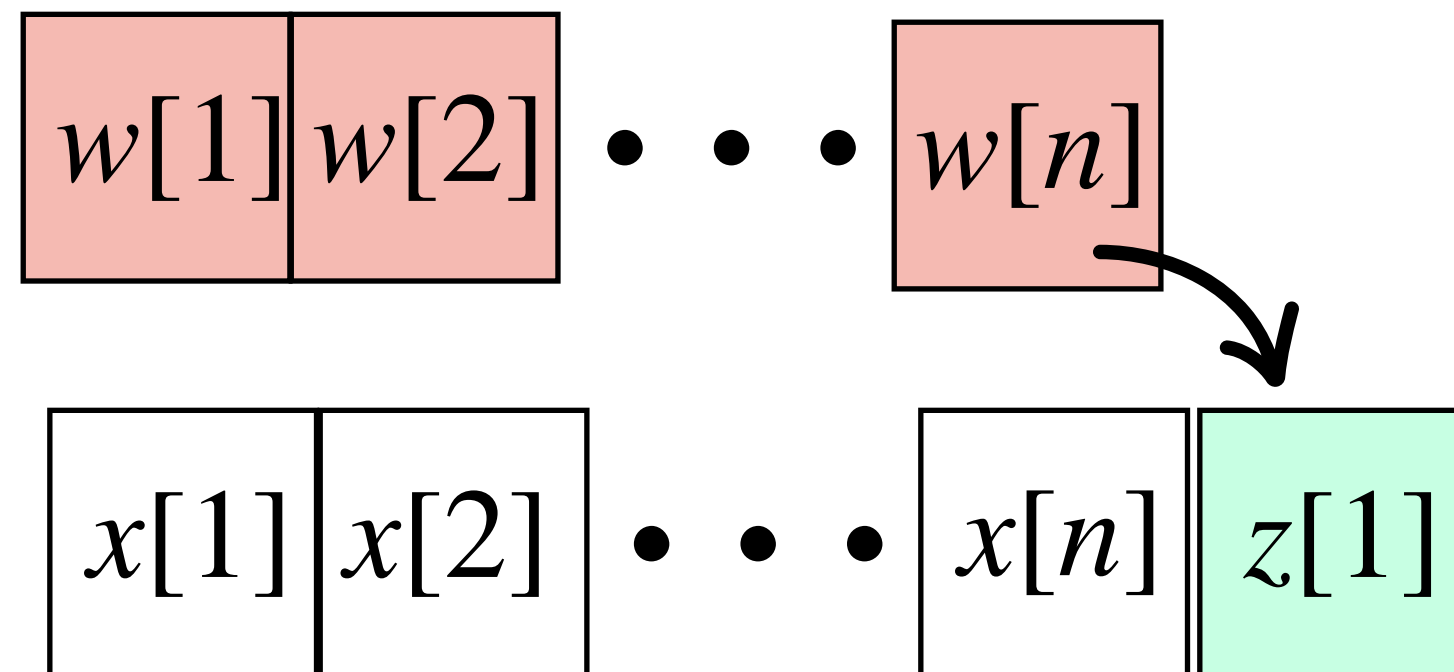


E2E Feedback: Hardness of Learning

- Learning from E2E data alone is computationally **hard**, even if next-token prediction problem for the function class is easy.

Autoregressive Linear Thresholds: $f_w(z) = \text{thr}(\langle w, z[-d :] \rangle)$

$$\mathcal{F}_{d,lin} = \{f_w : w \in \mathbb{R}^d\}$$

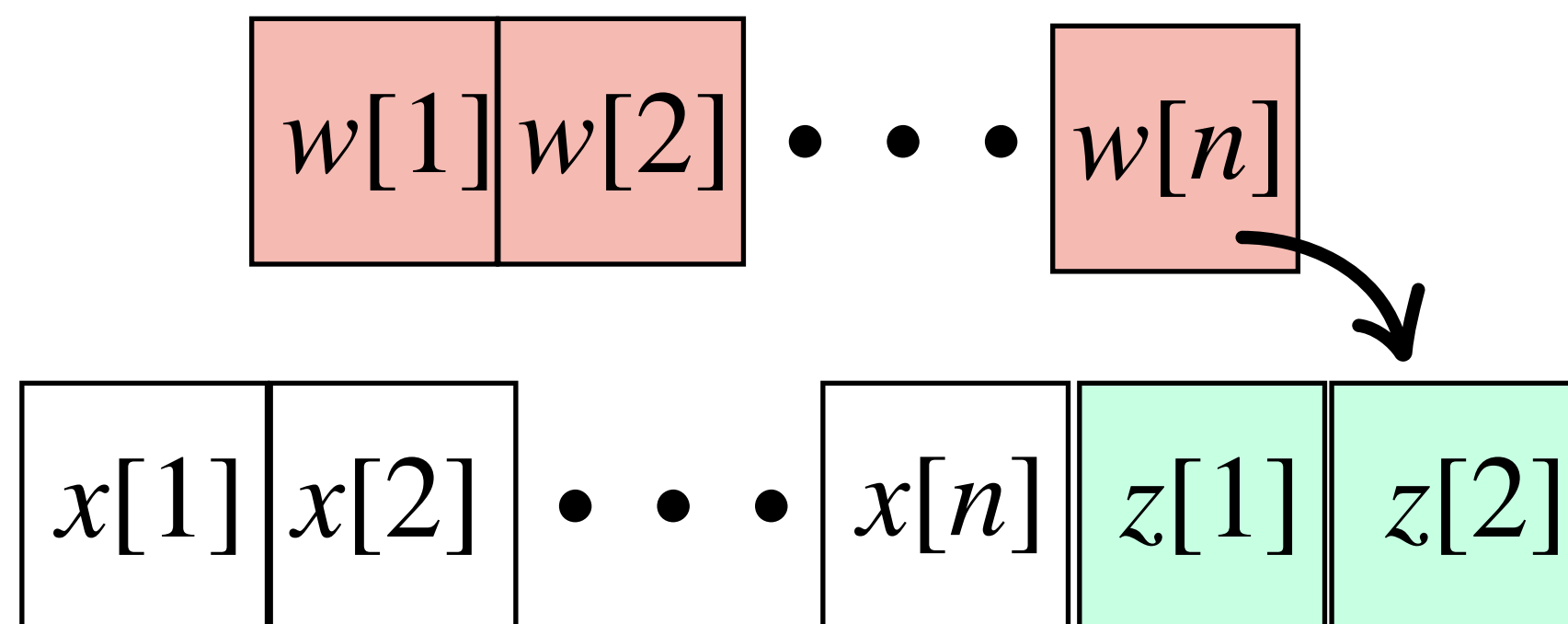


E2E Feedback: Hardness of Learning

- Learning from E2E data alone is computationally **hard**, even if next-token prediction problem for the function class is easy.

Autoregressive Linear Thresholds: $f_w(z) = \text{thr}(\langle w, z[-d :] \rangle)$

$$\mathcal{F}_{d,lin} = \{f_w : w \in \mathbb{R}^d\}$$

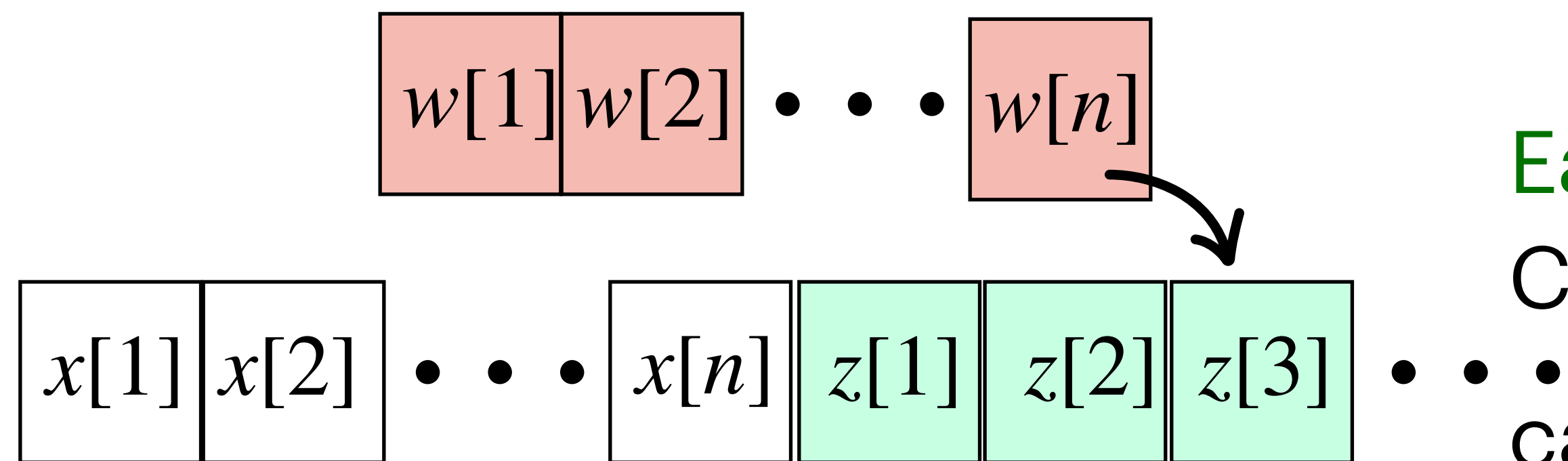


E2E Feedback: Hardness of Learning

- Learning from E2E data alone is computationally **hard**, even if next-token prediction problem for the function class is easy.

Autoregressive Linear Thresholds: $f_w(z) = \text{thr}(\langle w, z[-d :] \rangle)$

$$\mathcal{F}_{d,lin} = \{f_w : w \in \mathbb{R}^d\}$$



Easy to learn with CoT data
Consistency (i.e., next-token prediction)
can be solved by LP feasibility.

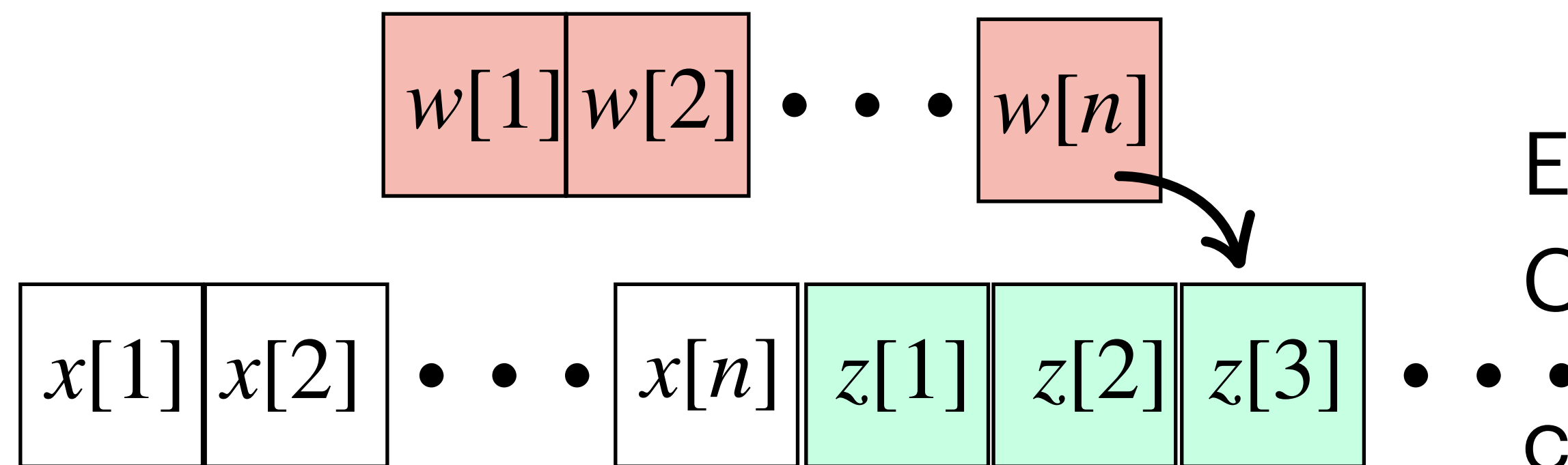
$\mathcal{F}_{d,lin}^{e2e-T}$ is hard to learn without CoT data for: $T = \Theta(d)$ [Joshi et al. 25]

E2E Feedback: Hardness of Learning

- Learning from E2E data alone is computationally **hard**, even if next-token prediction problem for the function class is easy.

Autoregressive Linear Thresholds: $f_w(z) = \text{thr}(\langle w, z[-d :] \rangle)$

$$\mathcal{F}_{d,lin} = \{f_w : w \in \mathbb{R}^d\}$$



Easy to learn with CoT data
Consistency (i.e., next-token prediction)
can be solved by LP feasibility

$\mathcal{F}_{d,lin}^{e2e-T}$ is hard to learn without CoT data for: $T = \omega_d(1)$ [Joshi et al. 26]

Open for $T = O(1)$

Attention and Transformers

- In practice, \mathcal{F} : the class of **transformers**
- Transformers have **parameter sharing**. The same “fixed-size” transformers is applied for all steps of generation.
- Autoregressive models allow: **Runtime** \ggg **# of parameters**.
- The number of parameters does not depend on the input sequence length $|x|$ or how many steps of thinking during inference, i.e., T .
- Transformers are **universal** in that they can **simulate arbitrary computations**.
- **Attention** instead of **Bounded memory**.

Transformer size \propto description length of the program,
generation length \propto runtime of the program [Joshi et al. 25].

Summary

- **Time invariance** allows decoupling of the **number of parameters** from the **runtime of inference**.
- This is what allows **sample complexity** (nearly) **independent of generation length** in autoregressive models.
- **CoT significantly helps computationally**, making hard problems easy to learn by intermediate step-by-step supervision
- **Universality** of the learning paradigm of **CoT + Attention-based architecture**

Part II. Learning from Heterogeneous Data in Pre-training

Heterogeneity, Stochasticity and Coverage

Data Heterogeneity: Non-Unique CoTs

- Often CoTs are **non-unique**. One question may have many valid reasonings
- CoT data collected is typically **heterogeneous**, with **diverse** thinking styles

data/CoT/gsm_train.json

Prose derivation

```
First find the total number of calories in the
pancakes: 6 pancakes * 120 calories/pancake =
720 calories
Then find the total number of calories in the
bacon: 2 strips * 100 calories/strip = 200
calories
```

```
Jerry's breakfast includes 6 pancakes with 120
calories each, two strips of bacon with 100
calories each, and a bowl of cereal with 200
calories. How many calories is his breakfast
total?
```

data/PoT/numglue.json

Python code

```
connie = 323
comparison = 175

juan = comparison + connie

print(juan)
```

```
Connie has 323 marbles. Juan has
175 more marbles than Connie. How
many marbles does Juan have?
Please respond by writing a
program in Python.
```

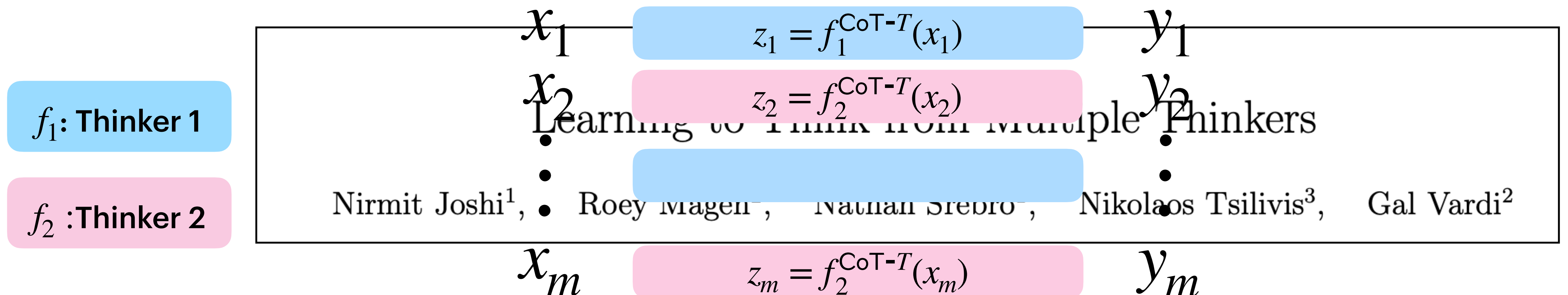
Setting: Learning from Multiple Thinkers

- Multiple generators (i.e., thinkers) (f_1, \dots, f_κ) that represent the same concept end-to-end.

$$f_1^{CoT-T} \equiv \dots \equiv f_\kappa^{CoT-T} \equiv h_\star$$

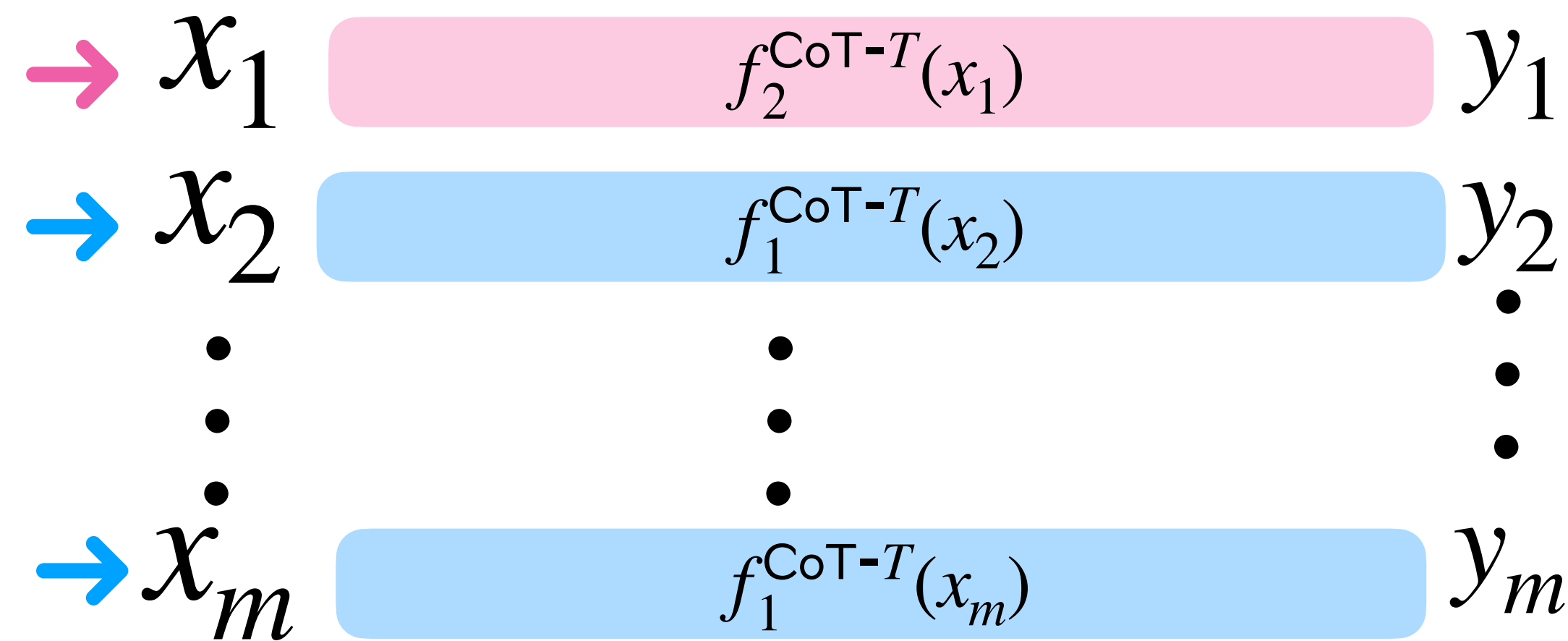
Computational hardness [Joshi et al. 26]: Learning from multiple thinkers is **Crypto-Hard** in **passive data collection settings**, even though it was easy if a single thinker contributed all the CoTs.

- E.g., Release E2E training set for collection; thinkers contribute arbitrarily.



Positive Result: Active Data Collection

The learner queries for CoTs from (f_1, \dots, f_K) .



One thinker only gives you M_* examples.
(No over-exploitation)

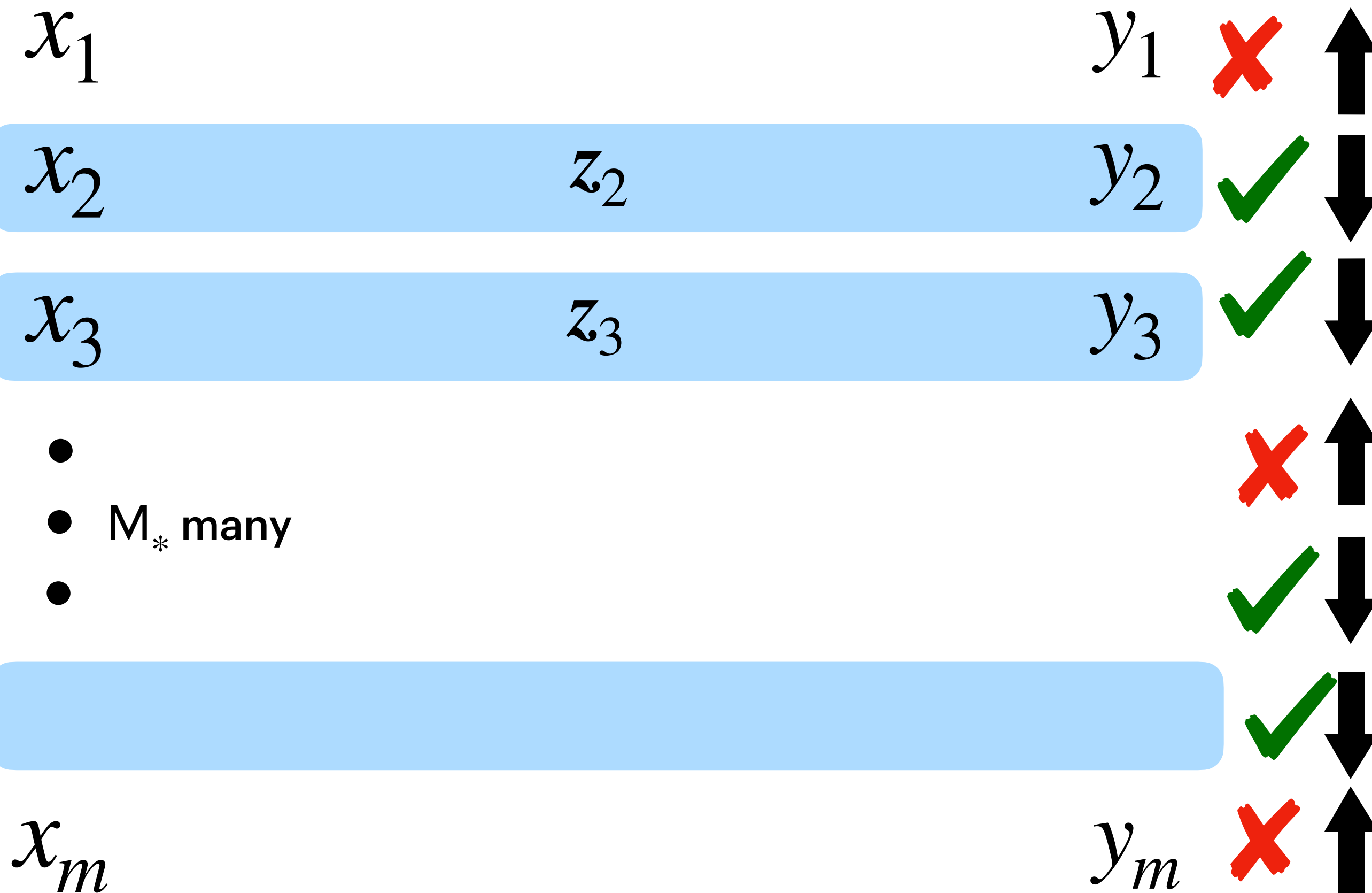
Q: Can we learn from
heterogeneous CoT sources?

Yes, we can “efficiently” learn a model with $1 - \epsilon$ accuracy using:

- $M_* = O(\text{VCdim}(\mathcal{F}) \log T)$ (Independent of ϵ)
- From $\kappa = O(\text{VC}(\mathcal{F}) \log T \log(1/\epsilon))$ thinkers (Active)
- E2E data $m^{\text{e2e}} = \tilde{O}(T \text{VC}(\mathcal{F}) / \epsilon)$ (Passive)

Key Intuition: Boosting

$$D_0 = \text{Unif}([m])$$



-
- M_* many
-

1. Get CoT from any **single** thinker.
2. CoT-Consistent / ERM on sampled points.
3. See which **E2E examples the predictor is erring**. Evaluate,

$$\epsilon_i = 1_{\{\hat{f}_1^{e^{2e^{-T}}}(x_i) \neq y_i\}}$$

This predictor is NOT perfect, since M_* is small, but it is a Weak learner.

4. **Upweight** and **downweight** examples where the predictor errs / is accurate, respectively.

Key Intuition: Boosting

$$D_0 = \text{Unif}([m]) \rightarrow D_1 \dots$$

$$x_1 \quad z_1 \quad y_1$$

$$x_2 \quad y_2$$

$$x_3 \quad y_3$$

-
- M_* many
-

$$x_m \quad z_m \quad y_m$$

1. Get CoT from any **single** thinker.
2. CoT-Consistent / ERM on sampled points.
3. See which **E2E examples the predictor is erring**. Evaluate,

$$\epsilon_i = 1_{\{\hat{f}_1^{e^{2e^{-T}}}(x_i) \neq y_i\}}$$

This predictor is NOT perfect, since M_* is small, but it is a Weak learner.

4. **Upweight** and **downweight** examples where the predictor errs / is accurate, respectively.
5. Iterate

$O(\log(1/\epsilon))$ rounds suffice to learn an aggregated predictor with accuracy $1 - \epsilon$.

Interpretations of Algorithm

- **Autocurriculum** [Rajaraman et al., 2026]: even if a single thinker provides CoT, we need only $\propto \log 1/\epsilon$ (active) CoTs instead of $1/\epsilon$ (passive). The algorithm self-decides which examples to collect CoTs.
- **Model Aggregation** [Joshi et al., 2026]: we can learn from traces that are completely different, and one thinker has only a small contribution to the training set.
- **Iterative and Adaptive Model Fine Tuning** rounds of fine-tuning on different sources. In a few rounds, we can have highly accurate models.

Learning to Reason with Curriculum I:
Provable Benefits of Autocurriculum

Nived Rajaraman*

Audrey Huang[†]

Miro Dudik[‡]

Robert Schapire[§]

Dylan J. Foster[¶]

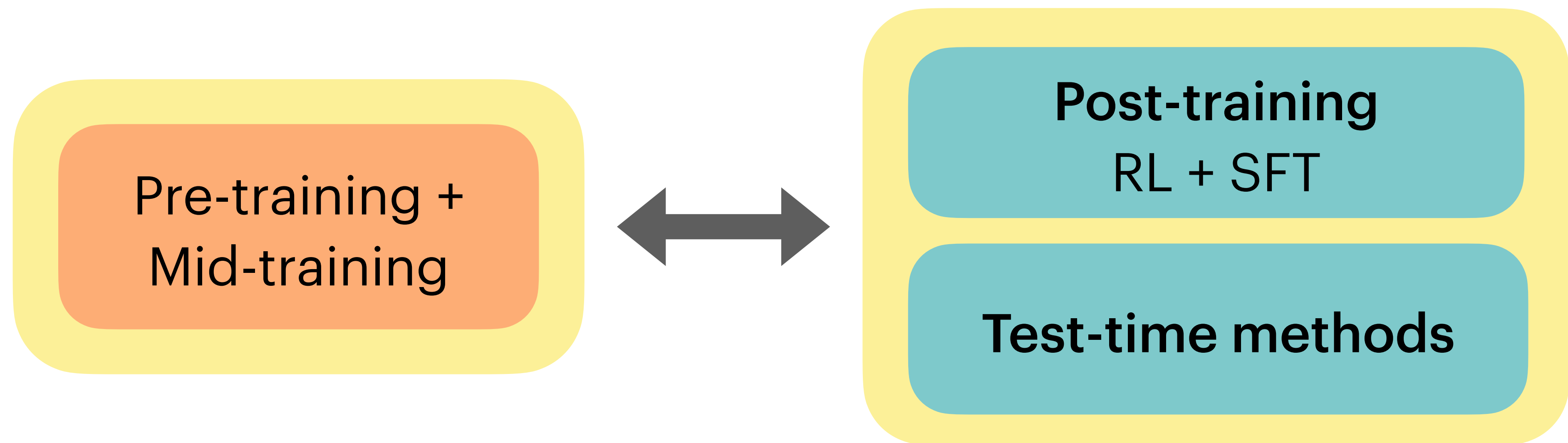
Akshay Krishnamurthy^{||}

Here Weak Learner \equiv One round of CoT Collection and Fine-tuning

5 minute break

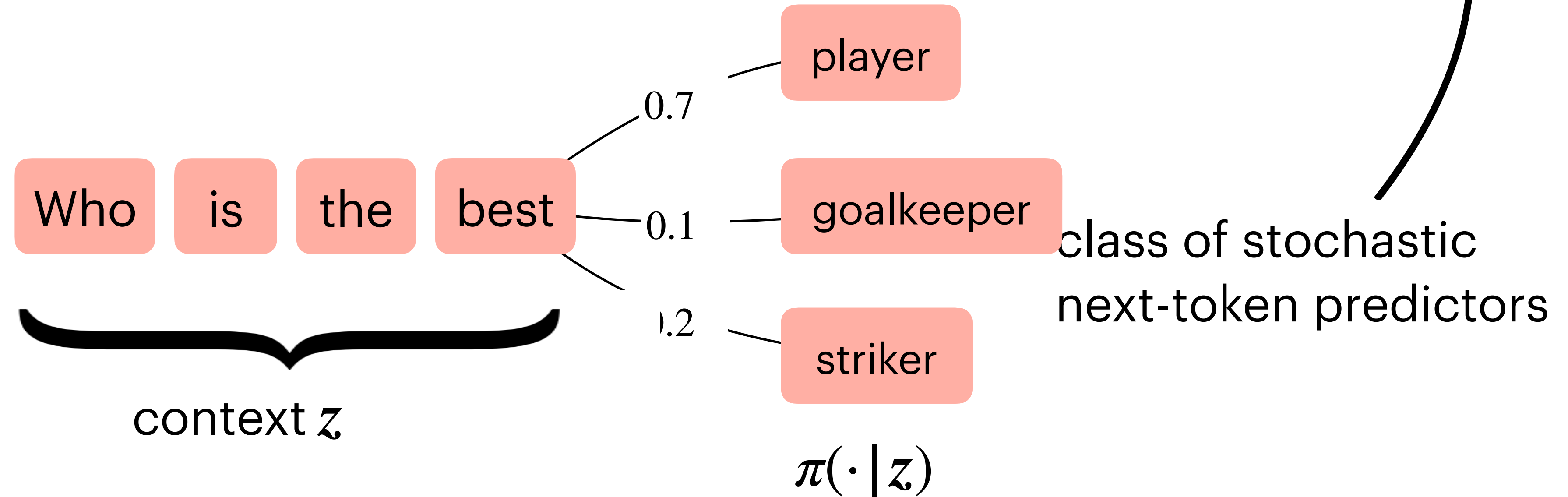
How do Pre-training and Downstream Post-training Interact?

- E2E learning is hard without assumptions [Joshi et al., 2025] and RL is only harder.
- In practice, models are pre-trained / mid-trained / SFT'ed prior to RL (warm-start).



Stochastic models

Heterogeneous pre-training data can be viewed as coming from a stochastic policy $\pi_{\star} \in \Pi$



$\pi^{\text{CoT-}T}(\cdot | \mathbf{x})$: distribution over length- T CoTs induced by π on prompt x

$\pi^{\text{e2e-}T}(\cdot | \mathbf{x})$: distribution over answers on the prompt x

Setting: Pre-training a stochastic model

Downstream verifier. Let $r^\star(\mathbf{x}, y)$ denote a ground-truth *outcome verifier*,

$$r^\star(\mathbf{x}, y) = \mathbf{1}(y \text{ is a correct answer for } \mathbf{x})$$

Denote accuracy of $\hat{\pi}$ (i.e., prob. of correct answers):

$$\text{Acc}(\hat{\pi}) = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}, y \sim \hat{\pi}^{e^{2e^{-T}}(\cdot|\mathbf{x})}} [r^\star(\mathbf{x}, y)]$$

Simplifying assumption: $\text{Acc}(\pi_\star) = 1$ for $\pi_\star \in \Pi$. (π_\star predicts answers correctly).

Special case: π_\star realized by mixture of multiple thinkers

Thinkers agree on final answer, so $r^\star(\mathbf{x}, y) = \mathbf{1}(y = y_\mathbf{x}^\star)$ ($y_\mathbf{x}^\star =$ correct answer)

Furthermore, $\pi_\star^{\text{CoT}-T}(\cdot|\mathbf{x}) = \frac{1}{k} \sum_{i=1}^k \delta_{z_i}$ where $z_i = f_i^{\text{CoT}-T}(\mathbf{x})$

Setting: Pre-training a stochastic model

Learner gets a heterogeneous dataset of CoTs from π_\star ,
 $\{(\mathbf{x}_i, \mathbf{z}_i)\}_{i=1}^n$ where $\mathbf{x}_i \sim \mathcal{D}$ and $\mathbf{z}_i \sim \pi_\star^{\text{CoT}-T}(\cdot | \mathbf{x}_i)$

Natural approach: Pre-training via MLE / next-token prediction.

$$\hat{\pi}_{\text{MLE}} \leftarrow \arg \min_{\pi \in \Pi} \sum_{i=1}^n \sum_{t=1}^T -\log(\pi(z_{i,t} | \mathbf{x}_i, \mathbf{z}_{i,\leq t-1}))$$

- MLE optimizes the same cross-entropy loss typically used in pre-training/SFT.
- MLE attempts to match distributions, **agnostic to downstream verifier.**

What makes a model “post-trainable”?

Accuracy = $\hat{\pi}$ generates the correct answer in one attempt (pass@1).

Coverage $\approx \hat{\pi}$ generates a correct CoT in a few attempts (pass@N).

$$\text{Acc}(\hat{\pi}) = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}, y \sim \hat{\pi} e^{2e-T(\cdot|\mathbf{x})}} [r^*(\mathbf{x}, y)] \quad \Bigg| \quad \text{Cov}_N(\hat{\pi}) = \Pr \left(\frac{\pi_{\star}^{\text{CoT}-T}(\mathbf{z} | \mathbf{x})}{\hat{\pi}^{\text{CoT}-T}(\mathbf{z} | \mathbf{x})} \geq N \right)$$

$\text{Cov}_N(\hat{\pi}) \approx$ mass of problems without correct CoTs in N attempts.

$\text{Cov}_N(\hat{\pi})$ is small $\implies \hat{\pi}$ is suitable for Best-of-N as an inference-time method.

(A weaker notion of coverage bounding outcome density ratio b/w $\pi_{\star}^{e^{2e-T}}$ and $\hat{\pi}^{e^{2e-T}}$ suffices)

$\text{Cov}_N(\hat{\pi})$ is small $\implies \hat{\pi}$ is suitable for improvement via RL (we’ll come back to this).

Coverage Emerges from Pre-training

Pre-training via MLE /next-token prediction can achieve good coverage even in sample regimes where achieving high accuracy is hard.

- **Coverage is easier to achieve than accuracy** [Foster et al. 2024, Chen et al., 2025],

$$1 - \text{Acc}(\hat{\pi}_{\text{MLE}}) \lesssim \frac{C_{\text{fine}}(\Pi)}{n}$$

$$\text{Cov}_N(\hat{\pi}_{\text{MLE}}) \lesssim \frac{C_{\text{fine}}(\Pi)}{n \cdot \log(N)} + \frac{C_{\text{coarse}}(\Pi)}{n}$$

covered mass
as N grows, this goes to 0

missing mass
MLE completely fails to
cover some mass of prompts

$C_{\text{fine}}(\Pi) \gg C_{\text{coarse}}(\Pi)$ are log-cov. numbers under log-loss at different scales.

Part III. RL Post-training and Beyond

From coverage to reasoning

Why coverage?

How does it enable downstream RL?

- Coverage bounds probability that $\hat{\pi}$ generates a correct CoT in N attempts.
- How does downstream RL sharpen the model to achieve high accuracy? (i.e., correct answer within a single attempt)

How does RL improve a **pre-trained model** π_{ref} satisfying:

$$\text{Cov}_N(\pi_{\text{ref}}) \leq \eta$$

And access to the exact ground-truth outcome verifier $r^*(\mathbf{x}, y)$.

RL via Expert Iteration

Natural Approach: Expert Iteration.

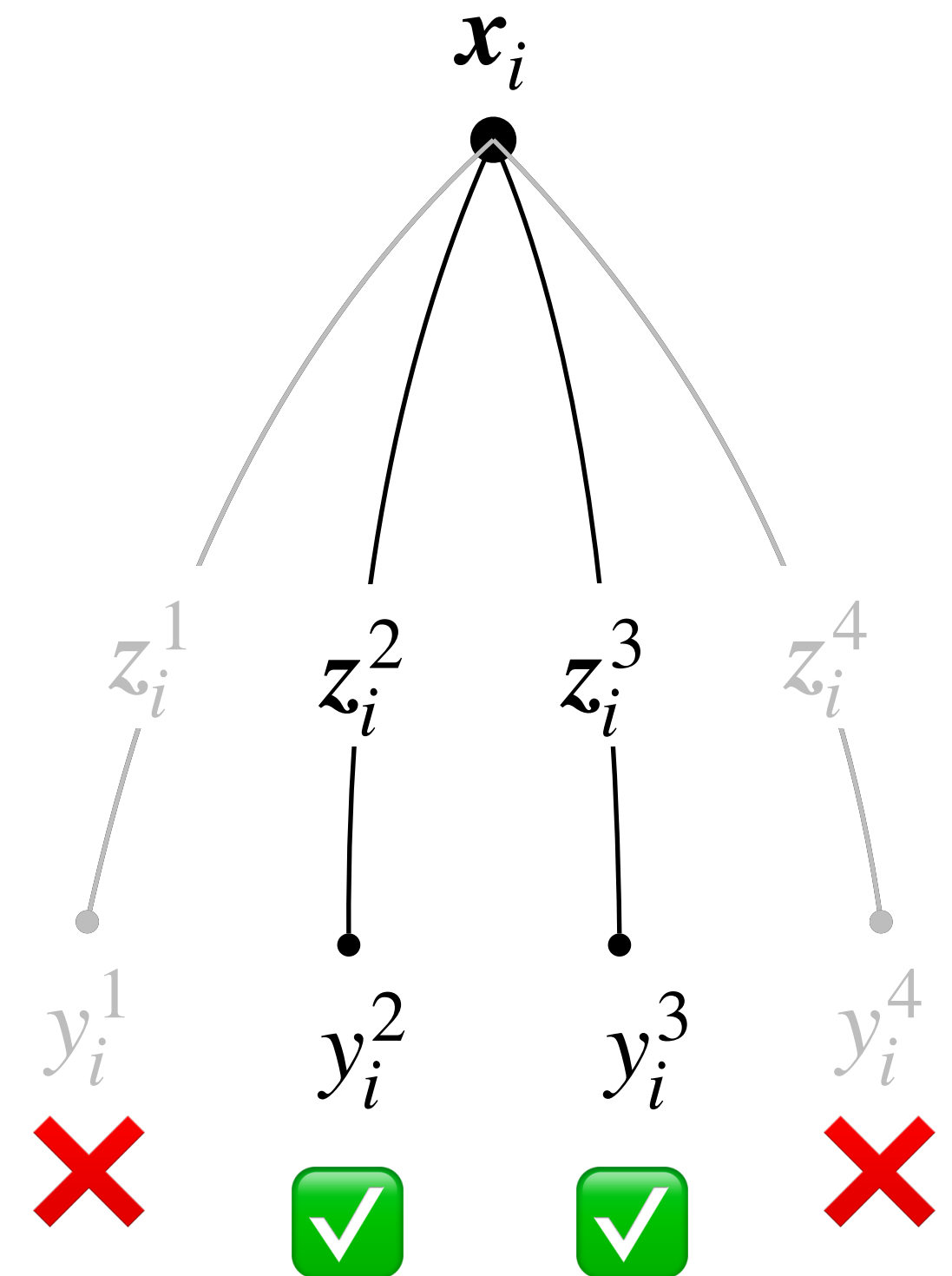
Given a dataset of n prompts $\{\mathbf{x}_i\}_{i=1}^n$,

1. Query π_{ref} for k CoTs on each \mathbf{x}_i , denoted $\mathbf{z}_i^1, \dots, \mathbf{z}_i^k$
(let the corresponding answers be y_i^1, \dots, y_i^k)
2. Discard CoTs \mathbf{z}_i^j with wrong answers: $r^\star(\mathbf{x}_i, y_i^j) = 0$.
3. Minimize the loss,

$$\hat{\pi}_{\text{ExpIt}} \leftarrow \arg \min_{\pi \in \Pi} \sum_{i=1}^n \sum_{j=1}^k \ell(\pi(\mathbf{z}_{i,k} | \mathbf{x}_i)) \cdot \underbrace{\mathbb{1}(r^\star(\mathbf{x}_i, y_i^j) = 1)}$$

Rollout cost: $O(kn)$ generations

Filter out CoTs with wrong answers

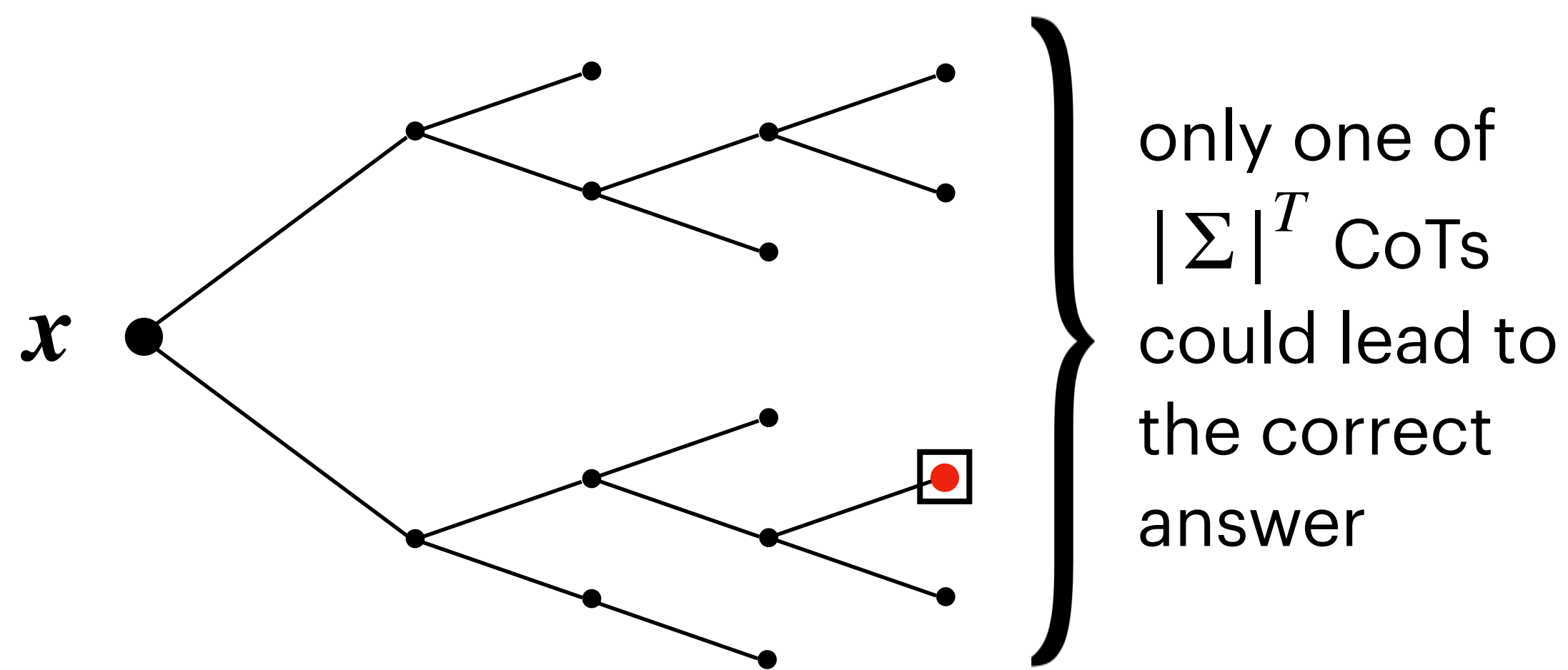


RL via Expert Iteration

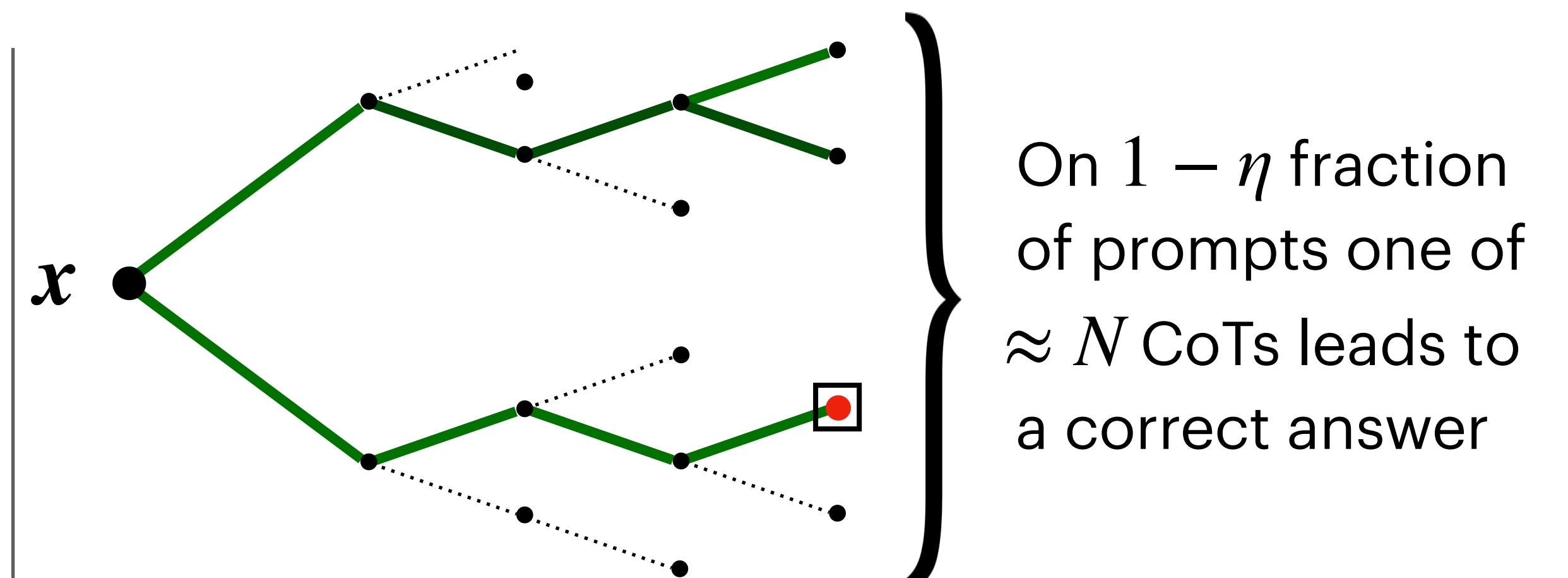
Expert Iteration [Rajaraman et al., 2026]. Suppose Π is deterministic and π_{ref} satisfies $\text{Cov}_N(\pi_{\text{ref}}) \leq \eta$. Up to log-factors,

$$1 - \text{Acc}(\hat{\pi}_{\text{ExpIt}}) \lesssim \frac{\text{VCdim}(\Pi) \cdot \log(TN)}{n} + \eta$$

Total number of CoT rollouts during training: $\tilde{O}(nN)$.



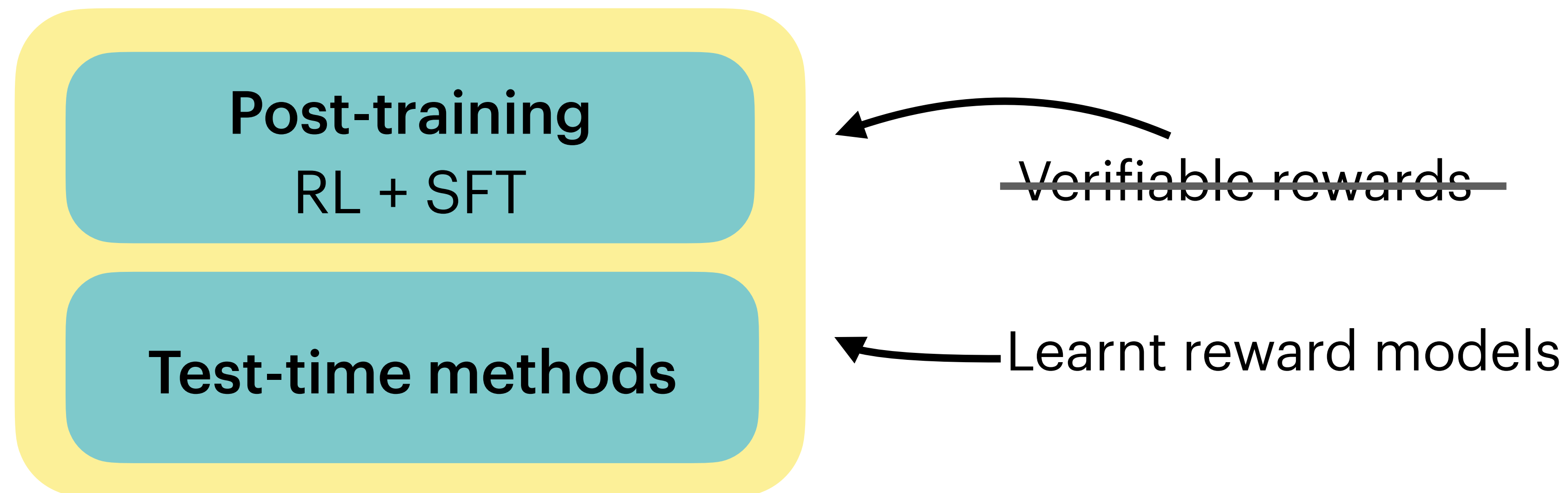
No pre-trained model



Given a pre-trained model π_{ref}

RL / BoN assumes verifiable rewards. What if we don't have them?

- **Across many domains:** formal math, code, assume access to a perfect reward
- In practice, even in these domains, reward hacking is possible
- **In many other domains (sciences):** we only have examples of desirable behavior



Approximate Verifiers: Reward Hacking

Noisy/learned verifiers \implies reward hacking (optimizing approximate rewards leads to poor performance under the true reward)

Several inference-time methods developed to address this issue.

Approximately correct reward models

[Huang et al., 2024]. Inference-time pessimism.

Pessimistic correction to rejection sampling to mitigate reward hacking as N grows.

Exact rewards + (very) approximate value models

[Rohatgi et al., 2025]. **VGB** (Value-Guided Sampling with Stochastic Backtracking)

Uses backtracking to avoid falling into “suboptimal valleys” while guiding the reasoning process using the approximate value model.

Learning Approximate Verifiers

So far our treatment focused on outcome/answer verifiers r^\star .

A large body of work on learning outcome verifiers from *preference feedback*.

New setting: **CoT Verifiers** [Balcan et al., 2025].

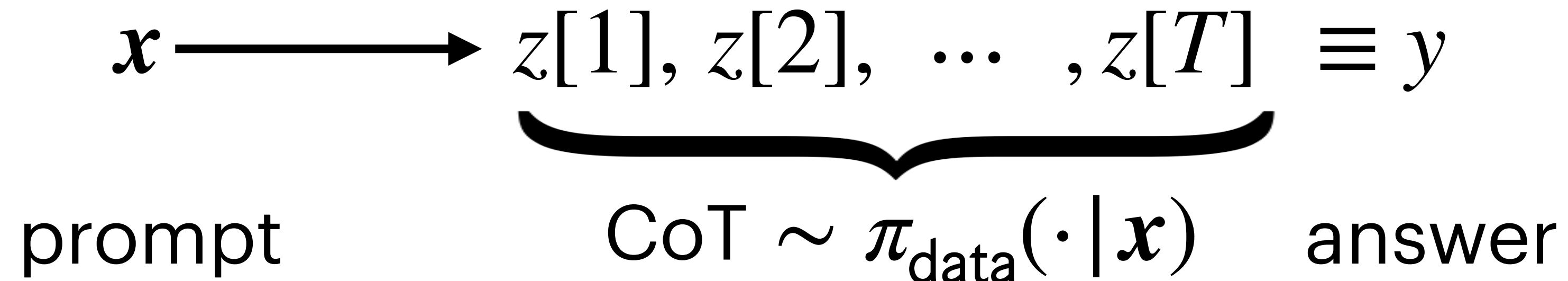
$$\begin{array}{ccc} \mathbf{x} & \longrightarrow & z[1], z[2], \dots, z[T] \equiv y \\ \text{prompt} & & \underbrace{\hspace{10em}}_{\text{CoT} \sim \pi_{\text{data}}(\cdot | \mathbf{x})} \quad \text{answer} \end{array}$$

Verifier $h \in \mathcal{H}$ labels length- t CoT by YES if all steps are “correct” according to it.

Critical in applications which need “monitorability”, e.g., AI safety [Guan et al., 2025].

Learning CoT Verifiers

New setting: **CoT Verifiers** [Balcan et al., 2025].



Learner gets a CoT dataset $\{(\mathbf{x}_i, \mathbf{z}_i)\}_{i=1}^n$ where $\mathbf{x}_i \sim \mathcal{D}$ and $\mathbf{z}_i \sim \pi_{\text{data}}(\cdot | \mathbf{x}_i)$.

+ labels from a true verifier $h^* \in \mathcal{H}$ identifying first wrong step of each \mathbf{z}_i .

Factors of 12: 1, 2, 3, 4, 6, 12, and of 8: 1, 2, 6, 10. So the GCD is 4

YES YES YES YES YES YES YES **NO**

Learning CoT Verifiers

Objective: Output a verifier \hat{h} such that the *two-sided error* is small

$$\Pr_{\mathcal{D}, \pi_{\text{data}}} \left[\hat{h} \text{ agrees with } h^* \text{ till first invalid step of } z \right] \geq 1 - \varepsilon$$

(Output YES on initial correct steps, and NO on the first wrong step)

Learning CoT Verifiers [Balcan et al., 2025]. Two-sided error ε achievable given a CoT dataset of size at least,

$$\text{VCDim}(\mathcal{H}) \cdot \frac{\log(T)}{\varepsilon}$$

Several extensions: “trustable” verification (must output NO on wrong CoTs)

Online learning of CoT verifiers [Balcan et al., 2026]

Part IV. Future Directions

Revisiting E2E and CoT feedback

Learning in structured settings when CoTs are hard to get

- Frontier model CoTs are often unavailable, even though we can query them for “answers” / summarized reasoning.
- In the worst case, E2E feedback is computationally hard to learn from.
How can we take advantage of it in structured settings?

Semiautomata: Compositional Tasks

Prompt: \mathbf{x}

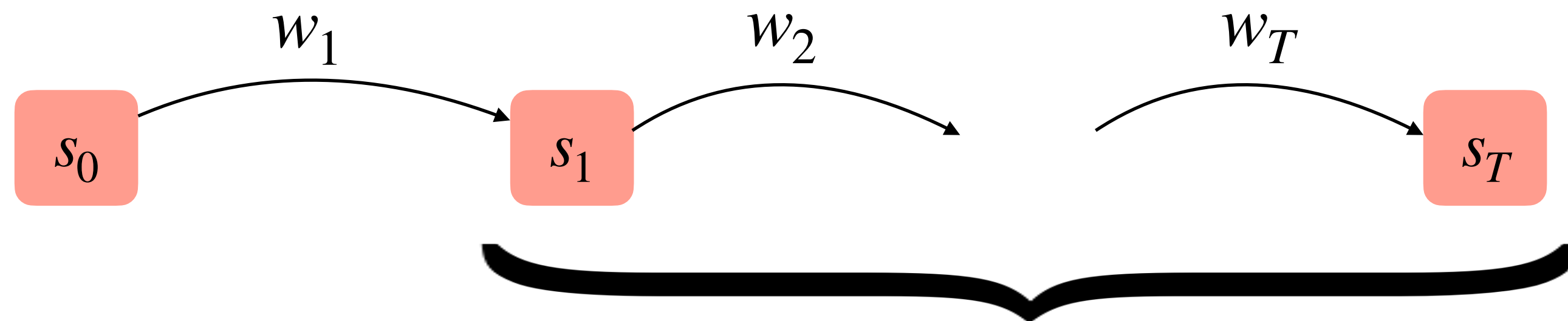
Initial state: s_0 +
 T "actions": w_1, \dots, w_T

"Current codebase state: $\langle A \rangle$
In the codebase, create new
functionality $\langle X \rangle$. Write
test-cases to check $\langle X \rangle$ works
as intended..."

s_0

w_1

w_2



$$f_{\star}^{\text{CoT-T}}(\mathbf{x}) = (s_1, \dots, s_T) \text{ where } s_t \leftarrow f_{\star}(s_{t-1}, w_t)$$

Learning objective:
Predict $s_T = f^{\text{e2e-T}}(\mathbf{x})$
(same as before)

- Every new state is generated using only the previous state and an input action
- Semiautomata capture compositional structure

Setting: Learning Semiautomata from E2E queries

Learner is given n prompts, $\{\mathbf{x}_i\}_{i=1}^n \sim_{iid} \mathcal{D}$

\mathcal{D} : **unknown** distribution over “length- T ” input prompts, $(s_0, \mathbf{w}_{1:T})$.

Can query f_\star on any input prompt \mathbf{x} (of arbitrary length t) to see $y = f^{e^{2e^{-t}}}(\mathbf{x})$.

Learning objective: accuracy $1 - \varepsilon$ under \mathcal{D} while minimizing #E2E queries.

Learning semiautomata [Rajaraman et al., 2026]. Accuracy $1 - \varepsilon$ achievable using an *ERM-oracle efficient curriculum-based* algorithm, making

$$2^{\tilde{O}(\sqrt{\log(T)})} \cdot \tilde{O}\left(\frac{\text{VCDim}(\mathcal{F})}{\varepsilon}\right)$$

E2E queries (and similar sample complexity)

Curriculum can help sidestep computational hardness in structured settings

Question. Is $\text{polylog}(T)$ dependence achievable?

A Deeper Understanding of How Pre-training influences Post-training

Beyond sequence-level coverage

- Sequence-level coverage is a sufficient condition for RL to improve the pre-trained model.
- In practice, RL can sometimes improve performance on problems where the initial model has 0% pass@N even for large N! (i.e., no coverage).
- How do we go beyond sequence-level coverage to understand how pre-training influences post-training and inference-time methods

New Learning Paradigms

Beyond E2E, CoT, and Verifier Feedback

- Modern training pipelines are more complicated than previously suggested.
- What are the right abstractions to understand newer paradigms: On-Policy Distillation (OPD), On-Policy Self-Distillation (OPSD) etc.