Chain of Thought and RL

13 Oct. 2025, Reasoning in AI

Scratchpads

- Teaching the model to provide rationale (e.g., computations) instead of just predicting the final answer.

SHOW YOUR WORK: SCRATCHPADS FOR INTERMEDIATE COMPUTATION WITH LANGUAGE MODELS

Maxwell Nye[1,2*] **Anders Johan Andreassen**[3] **Guy Gur-Ari**[3] **Henryk Michalewski**[2]

Jacob Austin[2] **David Bieber**[2] **David Dohan**[2] **Aitor Lewkowycz**[3] **Maarten Bosma**[2]

David Luan[2] **Charles Sutton**[2] **Augustus Odena**[2]

[1]MIT
[2]Google Research, Brain Team
[3]Google Research, Blueshift Team

```
Input:
2 9 + 5 7

Target:
<scratch>
2 9 + 5 7 ,  C: 0
2 + 5 , 6 C: 1  # added 9 + 7 = 6 carry 1
, 8 6 C: 0  # added 2 + 5 + 1 = 8 carry 0
0 8 6
</scratch>
8 6
```

Figure 2: Example of input and target for addition with a scratchpad. The carry is recorded in the digit following "C:". Comments (marked by #) are added for clarity and are not part of the target.

Chain of Thought

- LLMs can generate rationale without explicit training:
  - “Think step by step”
  - Few-shot CoT

Large Language Models are Zero-Shot Reasoners

Takeshi Kojima
The University of Tokyo
t.kojima@weblab.t.u-tokyo.ac.jp

Shixiang Shane Gu
Google Research, Brain Team

Machel Reid
Google Research*

Yutaka Matsuo
The University of Tokyo

Yusuke Iwasawa
The University of Tokyo

(a) Few-shot

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The answer is 8. ✗

(b) Few-shot-CoT

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4. ✓

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is

(Output) 8 ✗

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let’s think step by step.***

(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

Review

- Scratchpad/CoT improves both the expressivity and learning.
  - With a granular scratchpad, any poly-time algorithm can be taught to a LM.
  - We should keep the globality of our scratchpads low.
- Length generalization and its relation to positional embeddings.
  - Relative positional embeddings are often better than absolute ones.
  - However, they’re not enough and other ideas such as attention masking should be used.

Tool calling

Toolformer: Language Models Can Teach Themselves to Use Tools

Timo Schick **Jane Dwivedi-Yu** **Roberto Dessì**[†] **Roberta Raileanu**
Maria Lomeli **Luke Zettlemoyer** **Nicola Cancedda** **Thomas Scialom**
Meta AI Research [†]Universitat Pompeu Fabra

- Addition can be hard to learn, it's also not very efficient
- Let's just call a calculator!
- At training, we only need to exclude the output of the tool from the next-token prediction loss.
- At inference, we call the tool.
- Can increase efficiency and reliability.

The Nile has an approximate length of <API> **QA(What is the approximate length of the Nile?) → 6,853 km** </API> 6,853 kilometers, the White Nile being its main source.
If Venus had an atmosphere similar to Earth's then you would expect Venus' mean temperature to be 499 K (1.74 x 287) rather than 735 K which is <API> **Calculator(735 / 499) → 1.47** </API> 1.47 (735 / 499) times hotter than it should be.

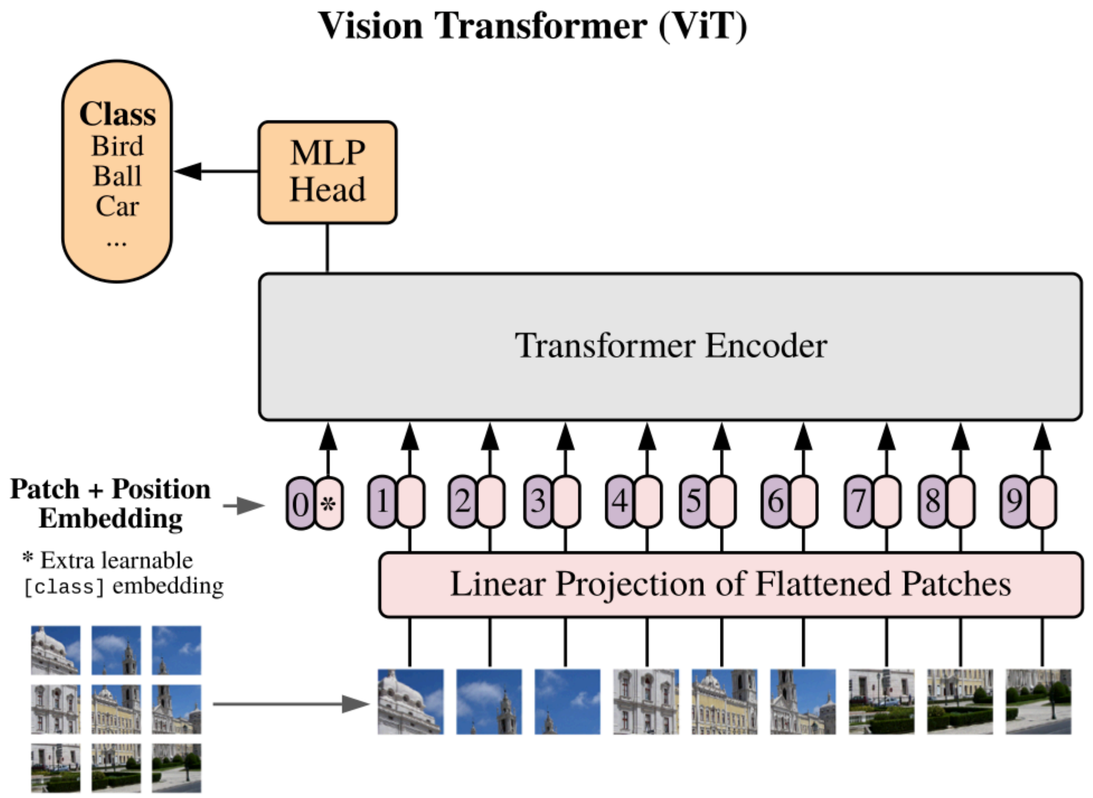
API Name	Example Input	Example Output
Question Answering	Where was the Knights of Columbus founded?	New Haven, Connecticut
Wikipedia Search	Fishing Reel Types	Spin fishing > Spin fishing is distinguished between fly fishing and bait cast fishing by the type of rod and reel used. There are two types of reels used when spin fishing, the open faced reel and the closed faced reel.
Calculator	27 + 4 * 2	35
Calendar	ε	Today is Monday, January 30, 2023.
Machine Translation	sûreté nucléaire	nuclear safety

Vision Transformers (ViT)

AN IMAGE IS WORTH 16X16 WORDS:
TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

**Alexey Dosovitskiy*,†, Lucas Beyer*, Alexander Kolesnikov*, Dirk Weissenborn*,
Xiaohua Zhai*, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer,
Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby*,†**
*equal technical contribution, †equal advising
Google Research, Brain Team
{adosovitskiy, neilhoulsby}@google.com

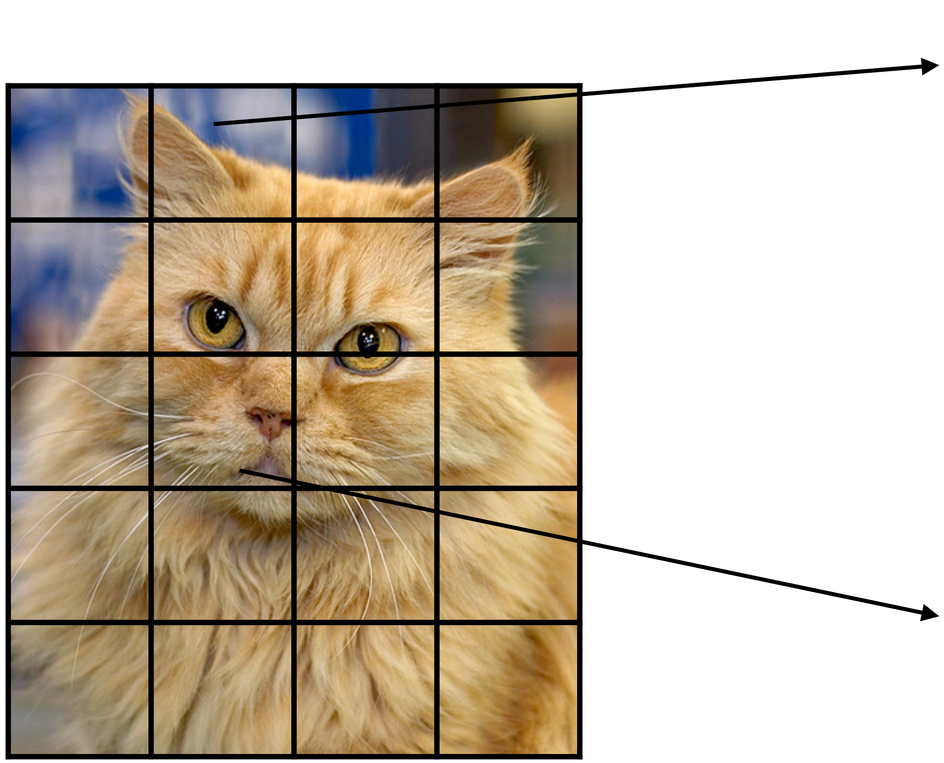
- Transformers can be also used on images.
- Each image is broken into *patches*, and patches are treated as tokens.
  - Often 14x14 or 16x16 patches.
- We often use an encoder-only model.
  - Having bidirectional attention.
- Using a classification token: <CLS> token



Most vision tasks are local

A few local features (e.g., patches) provide significant information on the output.

=> Common image classification is a *local* task.



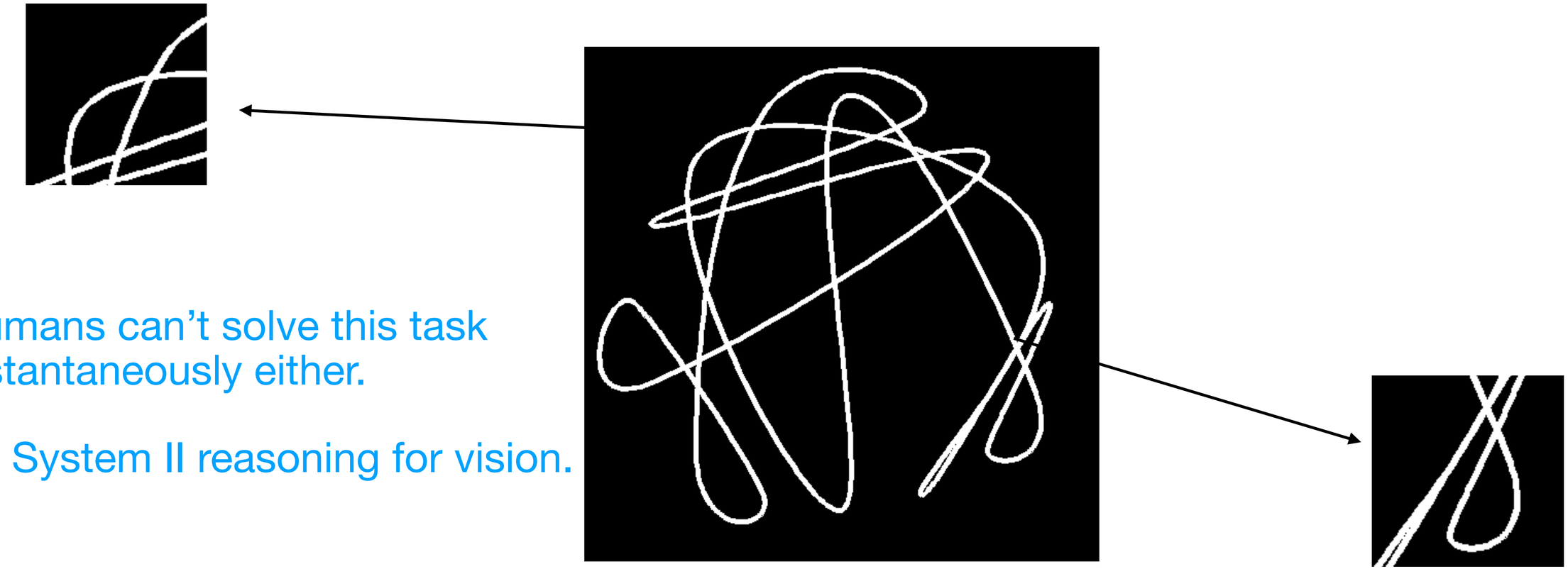
Probabilities given the two patches

Persian cat: 0.38
Egyptian cat: 0.15
tiger cat: 0.08
Siamese cat: 0.02
…

[Lotfi et al., 2024]

Not all visual tasks are local

How many strings are in the image? One or two?



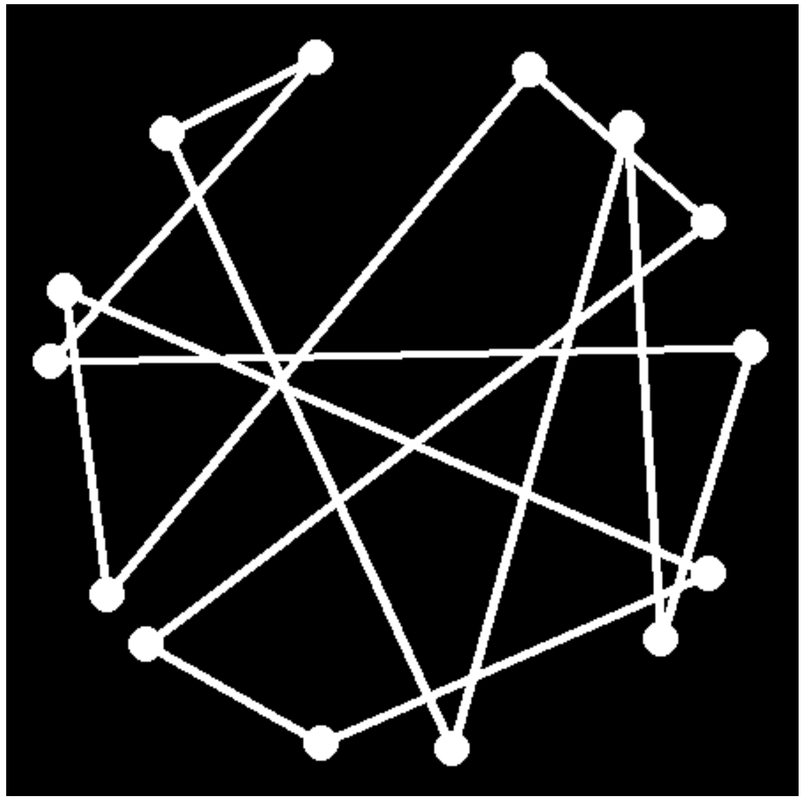
Humans can’t solve this task instantaneously either.

=> System II reasoning for vision.

Single patches are uninformative on the output —> global reasoning is needed.

[Lotfi et al., 2024]

More examples



Cycles task

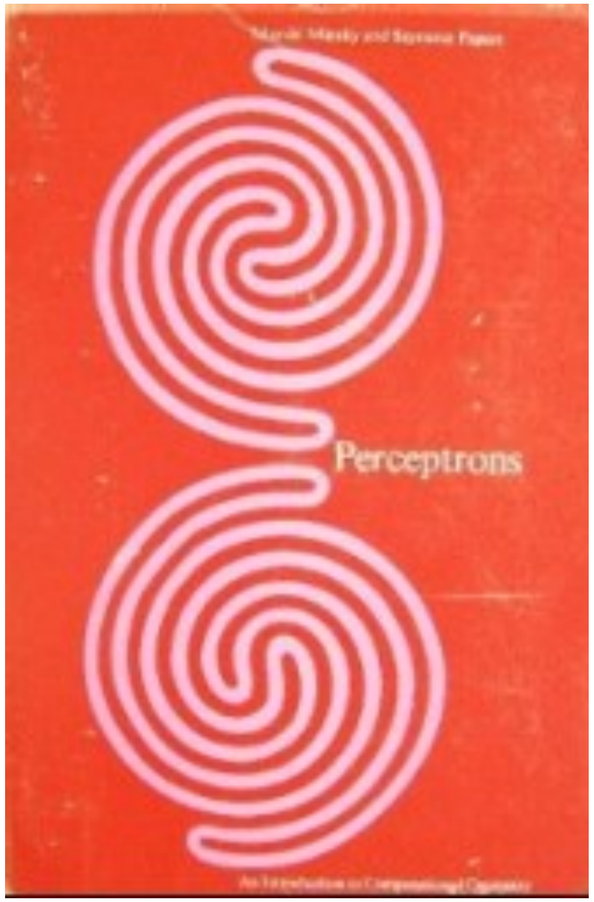
PVR task

[Lotfi et al., 2024]

Not all tasks are local

“Perceptrons can’t distinguish these figures.”

- *Current neural networks don’t have expressivity issues.*
  But, can they learn to distinguish such figures?



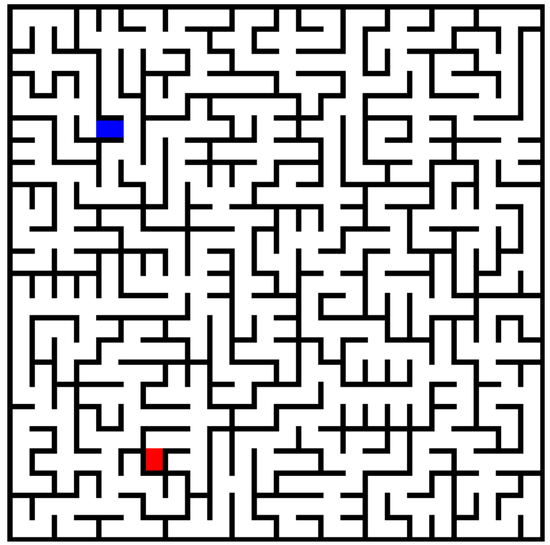
Perceptrons by M. Minsky & S. Papert, 1972

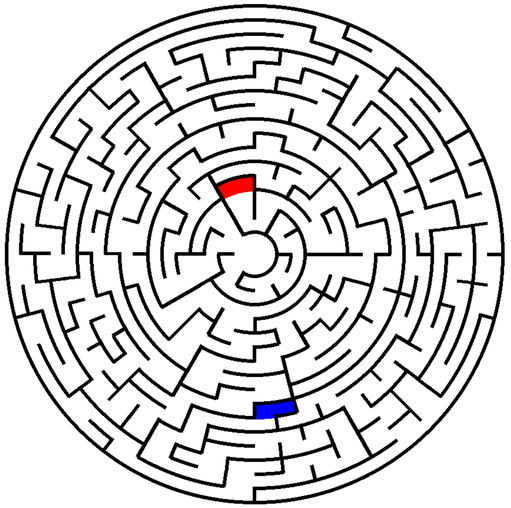
[Lotfi et al., 2024]

More examples

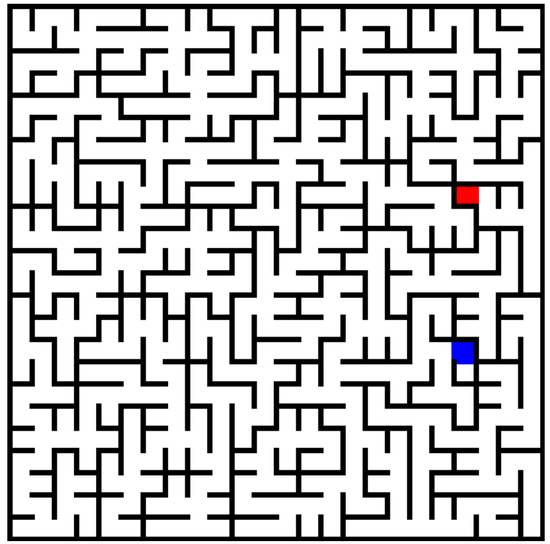
Are the two points connected?

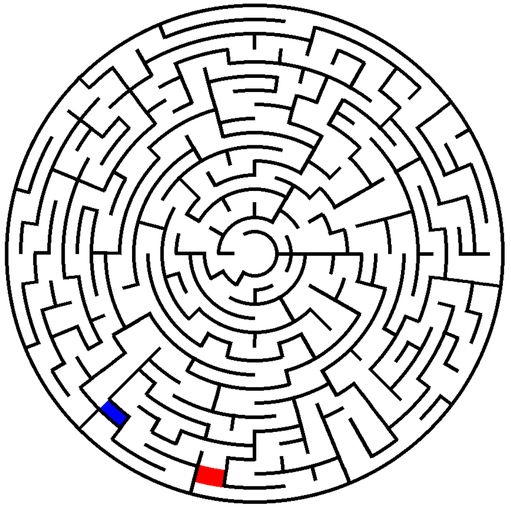
Yes





No

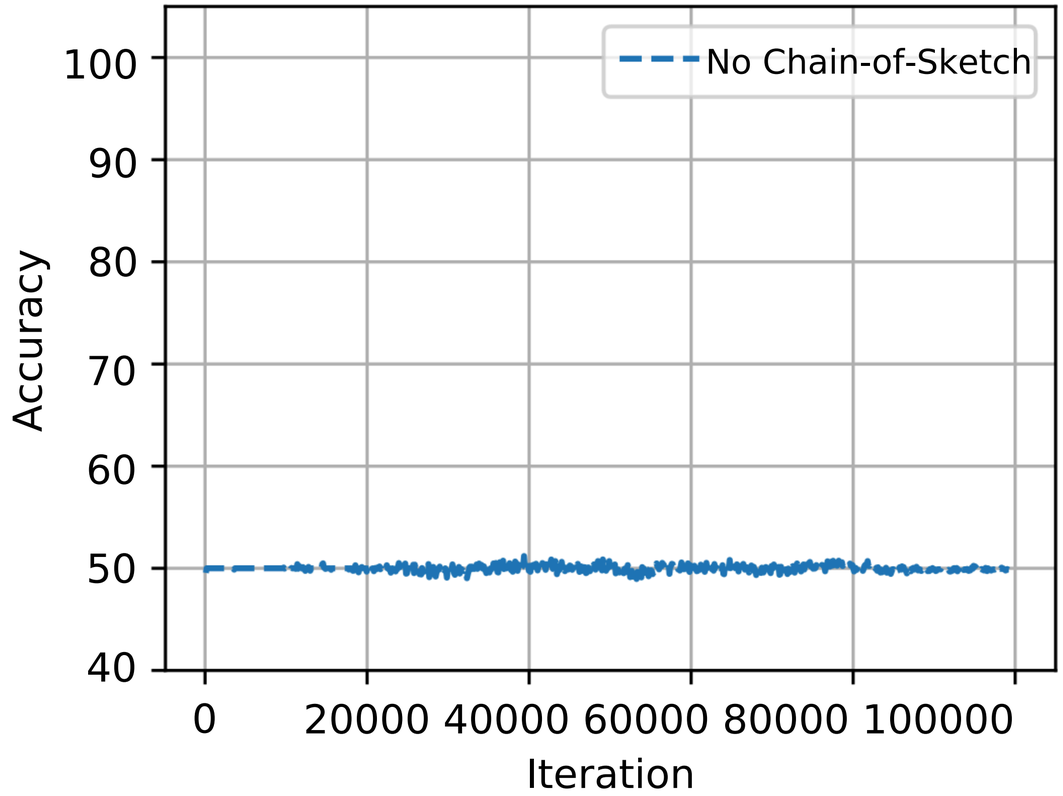




[Lotfi et al., 2024]

These tasks are not (efficiently) learnable

- E.g., training a ViT with 1M samples of strings task with 16 nodes:



- Using pre-trained models is not helpful.
- Issue: high globality

[Lotfi et al., 2024]

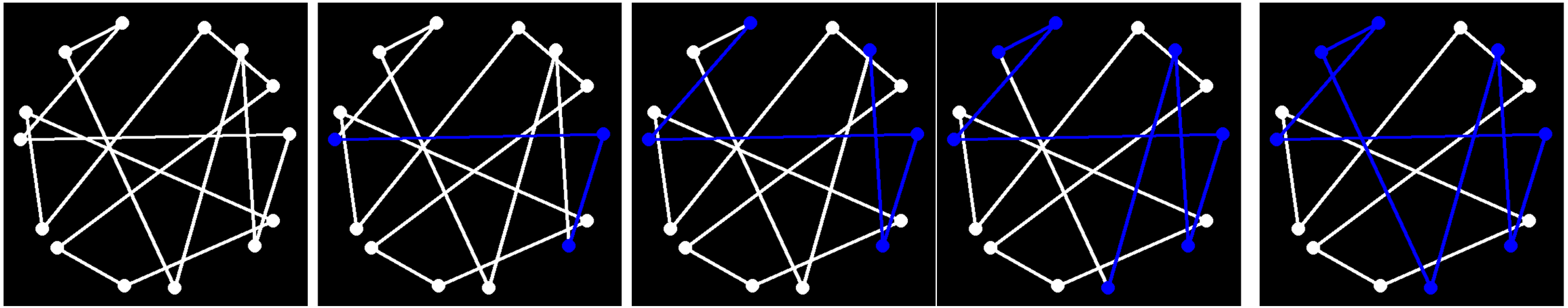
There tasks are also hard for SOTA multi-modal LLMs.

Task	Size	GPT-4o		o4-mini		o3	
		Prompt	ICL	Prompt	ICL	Prompt	ICL
Cycles	6 nodes	61.9%	58.9%	68.9%	62.2%	63.4%	62.1%
	8 nodes	51.6%	50.4%	55.4%	52.9%	54.7%	51.4%
	10 nodes	49.9%	48.8%	50.5%	48.4%	49.8%	51.3%
Maze (rect.)	4×4	50.6%	55.4%	55.3%	52.4%	52.4%	51.9%
	6×6	48.4%	51.8%	49.4%	51.3%	51.2%	48.6%

[Lotfi et al., 2024]

Solution: what would humans do?

- Classical (and local) vision tasks are often solved by humans instantly and intrinsically.
- Global tasks require thinking —> we may need to write something down or annotate the image, etc.

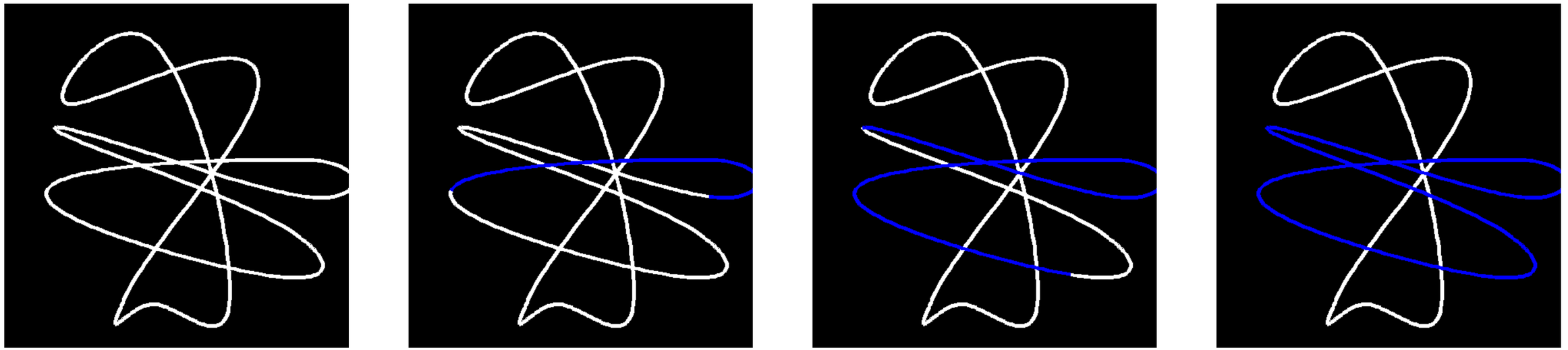


Chain-of-Sketch

[Lotfi et al., 2024]

Solution: what would humans do?

- Classical (and local) vision tasks are often solved by humans instantly and intrinsically.
- Global tasks require thinking —> we may need to write something down or annotate the image, etc.

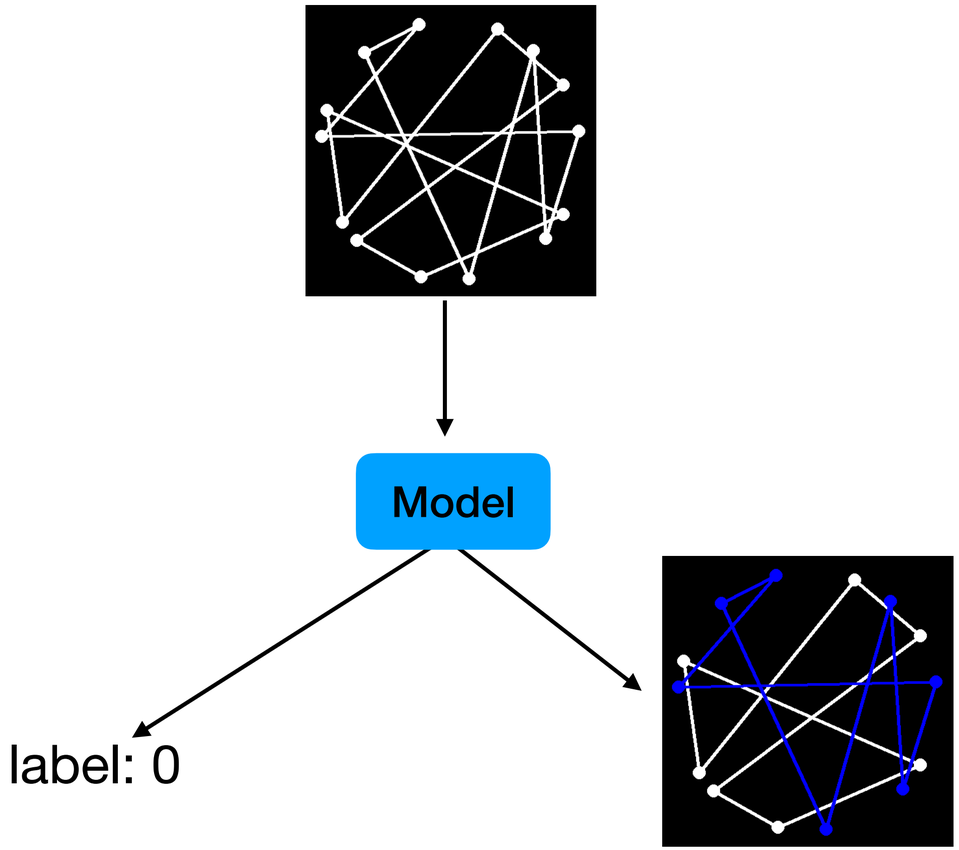


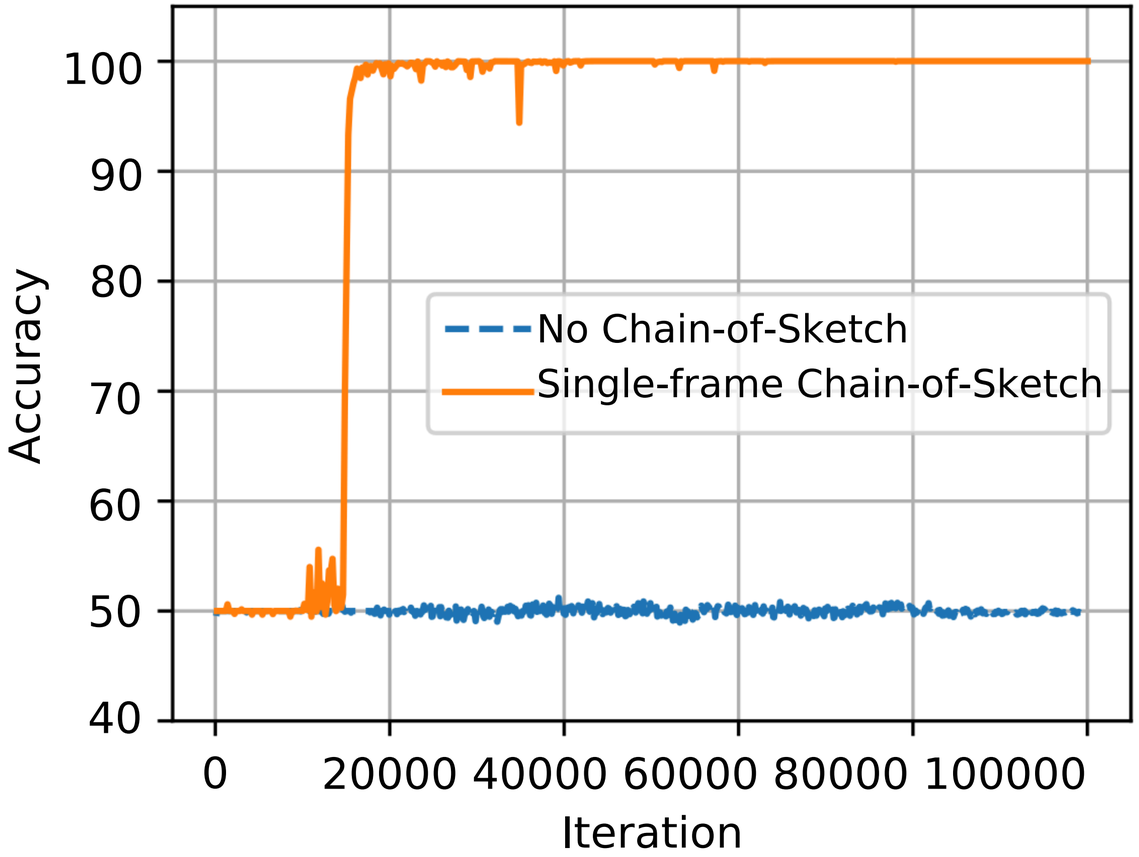
Chain-of-Sketch

[Lotfi et al., 2024]

Single-frame Chain-of-Sketch

Training the model to predict the underlying solution is helpful.

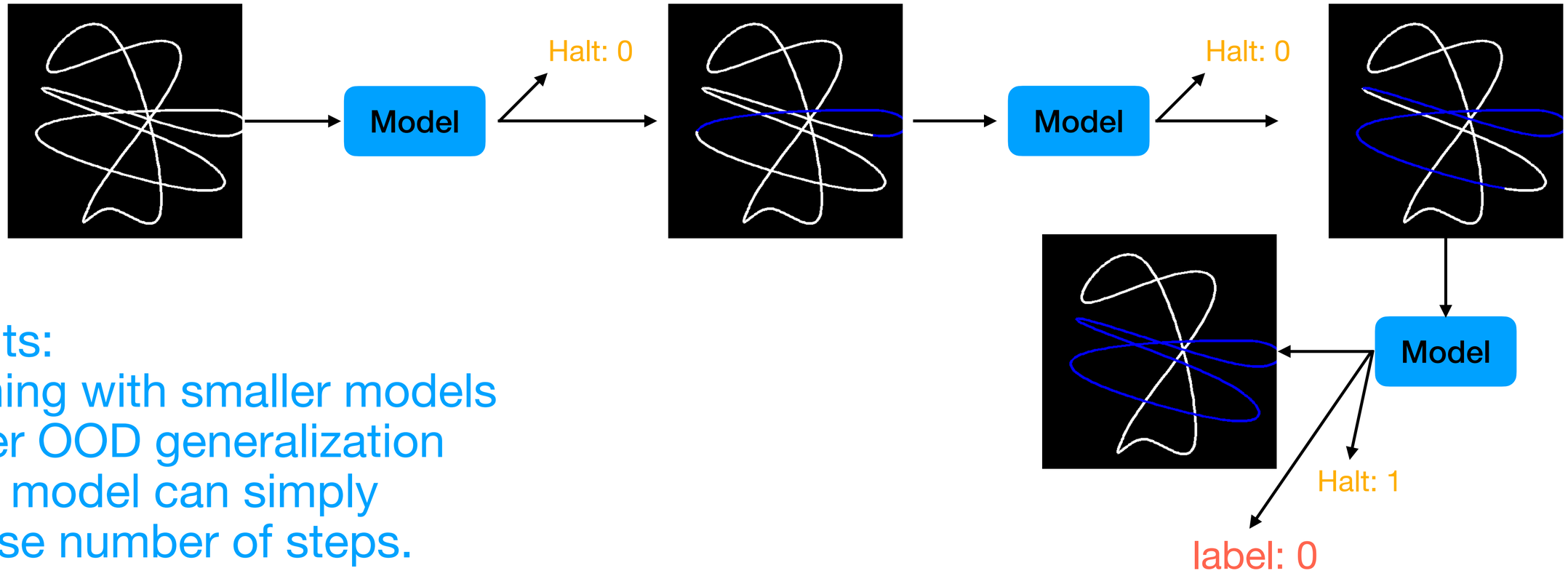




[Lotfi et al., 2024]

Inductive Chain-of-sketch

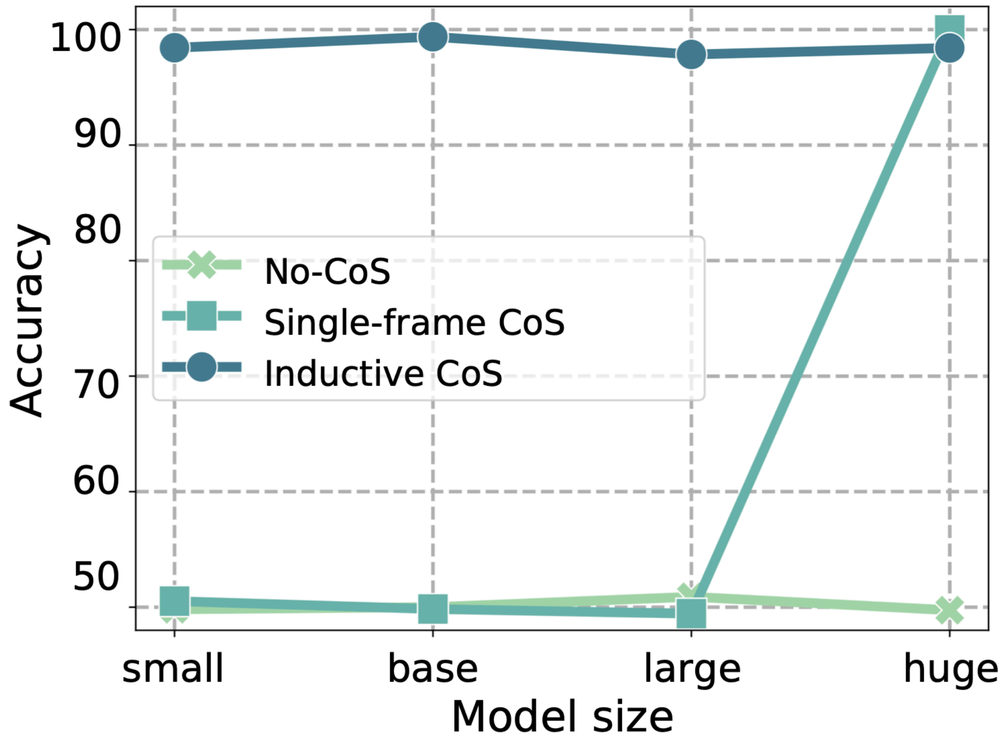
- Training the model to predict the underlying solution in multiple frames.
- Each time the model predicts the next frame using only the previous frame.



Benefits:
- learning with smaller models
- better OOD generalization
as the model can simply
increase number of steps.

[Lotfi et al., 2024]

Inductive Chain-of-sketch—benefits



Cycles 24

Dataset	Method	Accuracy (%)	
		ID	OOD
Maze 24 (Rect.)	single-frame CoS	**100.0**	54.4
	inductive CoS	99.8	**99.8**
PVR Grid 7x7	single-frame CoS	**100.0**	48.6
	inductive CoS	99.5	**82.2**

+ Learning with smaller models

+ Better OOD generalization as the model can simply increase number of steps.

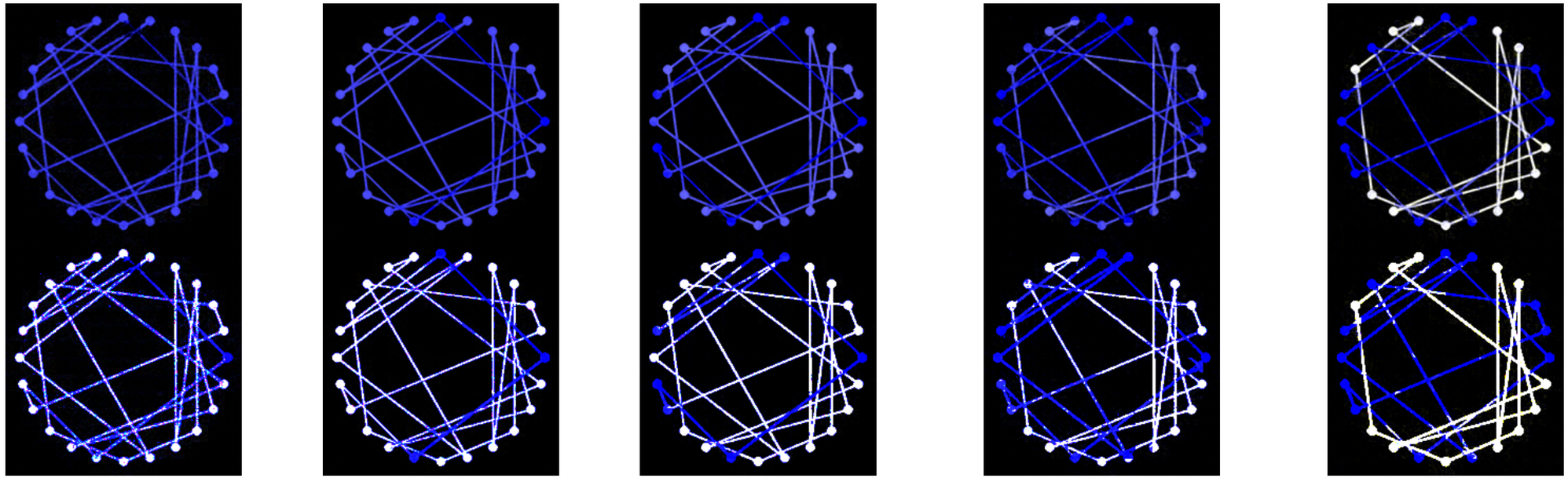
[Lotfi et al., 2024]

Hierarchical learning of the single-frame CoS

- We observed that the learning process of the scratchpad is hierarchical.

Model's output:

Output with higher contrast:



Iteration: 5k 10k 20k 25k 35k

- This is connected to the staircase phenomenon observed/proved when learning Boolean functions.

[E Abbe, E Boix, T Misiakiewicz '22 & '23]

[Lotfi et al., 2024]

Takeaway: We need to train models with high quality CoT data to improve their reasoning.

How to obtain such data?

Focus: Mathematical Reasoning

- There are datasets of math problems and solutions (with steps).

Problem: Beth bakes 4, 2 dozen batches of cookies in a week. If these cookies are shared amongst 16 people equally, how many cookies does each person consume?
Solution: Beth bakes 4 2 dozen batches of cookies for a total of 4*2 = <<4*2=8>>8 dozen cookies
There are 12 cookies in a dozen and she makes 8 dozen cookies for a total of 12*8 = <<12*8=96>>96 cookies
She splits the 96 cookies equally amongst 16 people so they each eat 96/16 = <<96/16=6>>6 cookies
Final Answer: 6

Problem: Mrs. Lim milks her cows twice a day. Yesterday morning, she got 68 gallons of milk and in the evening, she got 82 gallons. This morning, she got 18 gallons fewer than she had yesterday morning. After selling some gallons of milk in the afternoon, Mrs. Lim has only 24 gallons left. How much was her revenue for the milk if each gallon costs $3.50?
Mrs. Lim got 68 gallons - 18 gallons = <<68-18=50>>50 gallons this morning.
So she was able to get a total of 68 gallons + 82 gallons + 50 gallons = <<68+82+50=200>>200 gallons.
She was able to sell 200 gallons - 24 gallons = <<200-24=176>>176 gallons.
Thus, her total revenue for the milk is $3.50/gallon x 176 gallons = $<<3.50*176=616>>616.
Final Answer: 616

Problem: Tina buys 3 12-packs of soda for a party. Including Tina, 6 people are at the party. Half of the people at the party have 3 sodas each, 2 of the people have 4, and 1 person has 5. How many sodas are left over when the party is over?
Solution: Tina buys 3 12-packs of soda, for 3*12= <<3*12=36>>36 sodas
6 people attend the party, so half of them is 6/2= <<6/2=3>>3 people
Each of those people drinks 3 sodas, so they drink 3*3=<<3*3=9>>9 sodas
Two people drink 4 sodas, which means they drink 2*4=<<4*2=8>>8 sodas
With one person drinking 5, that brings the total drank to 5+9+8+3= <<5+9+8+3=25>>25 sodas
As Tina started off with 36 sodas, that means there are 36-25=<<36-25=11>>11 sodas left
Final Answer: 11

Figure 1: Three example problems from GSM8K. Calculation annotations are highlighted in red.

GSM8k [Cobbe et al. 2021]

MATH Dataset (Ours)

Problem: Tom has a red marble, a green marble, a blue marble, and three identical yellow marbles. How many different groups of two marbles can Tom choose?
Solution: There are two cases here: either Tom chooses two yellow marbles (1 result), or he chooses two marbles of different colors ($\binom{4}{2} = 6$ results). The total number of distinct pairs of marbles Tom can choose is $1 + 6 = \boxed{7}$.

Problem: If $\sum_{n=0}^{\infty} \cos^{2n}\theta = 5$, what is $\cos 2\theta$?
Solution: This geometric series is $1 + \cos^2\theta + \cos^4\theta + \cdots = \frac{1}{1-\cos^2\theta} = 5$. Hence, $\cos^2\theta = \frac{4}{5}$. Then $\cos 2\theta = 2\cos^2\theta - 1 = \boxed{\frac{3}{5}}$.

Problem: The equation $x^2 + 2x = i$ has two complex solutions. Determine the product of their real parts.
Solution: Complete the square by adding 1 to each side. Then $(x+1)^2 = 1 + i = e^{\frac{i\pi}{4}}\sqrt{2}$, so $x + 1 = \pm e^{\frac{i\pi}{8}}\sqrt[4]{2}$. The desired product is then $\left(-1 + \cos\left(\frac{\pi}{8}\right)\sqrt[4]{2}\right)\left(-1 - \cos\left(\frac{\pi}{8}\right)\sqrt[4]{2}\right) = 1 - \cos^2\left(\frac{\pi}{8}\right)\sqrt{2} = 1 - \frac{(1+\cos(\frac{\pi}{4}))}{2}\sqrt{2} = \boxed{\frac{1-\sqrt{2}}{2}}$.

MATH [Hendrycks et al. 2021]

- Issue: Limited size.
- Can we get more data?

Generating more data

Using the model itself

- Feature of (some) math datasets: having a final answer —> easy verification
  - Not true for theorems —> using formal math (e.g., Lean)
- How to generate data using the model?
  - Give questions to the model and generate solutions.
  - Check the final answers and keep the correct ones. (slightly error prone)
  - Fine tune the model on the correct solutions.

Generating more data

STaR

STaR: Self-Taught Reasoner
Bootstrapping Reasoning With Reasoning

Eric Zelikman[*1], **Yuhuai Wu**[*12], **Jesse Mu**[1], **Noah D. Goodman**[1]
[1]Department of Computer Science, Stanford University
[2] Google Research
{ezelikman, yuhuai, muj, ngoodman}@stanford.edu

Algorithm 1 STaR

Input M: a pretrained LLM; dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{D}$ (w/ few-shot prompts)

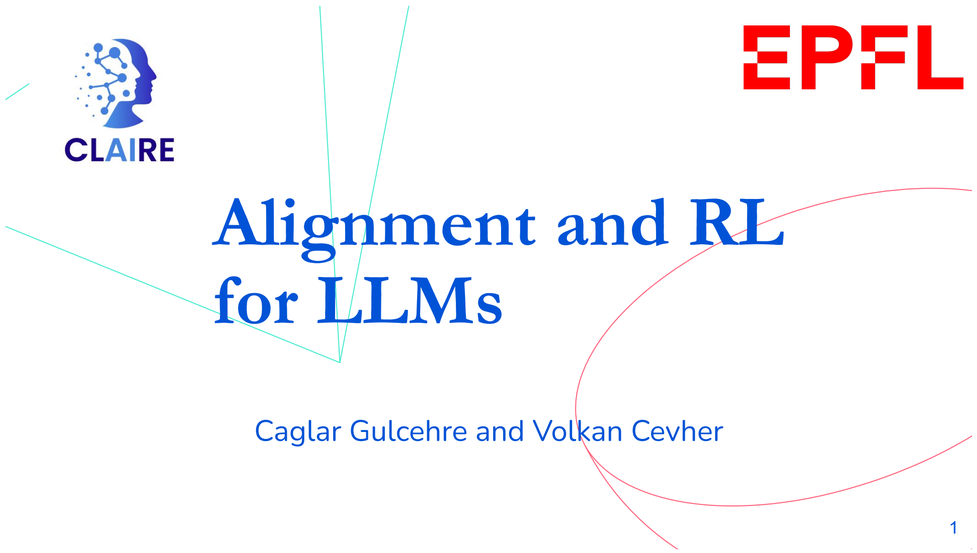
$M_0 \leftarrow M$ # Copy the original model
for n **in** $1...N$ **do** # Outer loop
  $(\hat{r}_i, \hat{y}_i) \leftarrow M_{n-1}(x_i) \quad \forall i \in [1, D]$ # Perform rationale generation
  $(\hat{r}_i^{\text{rat}}, \hat{y}_i^{\text{rat}}) \leftarrow M_{n-1}(\text{add\_hint}(x_i, y_i)) \quad \forall i \in [1, D]$ # Perform rationalization
  $\mathcal{D}_n \leftarrow \{(x_i, \hat{r}_i, y_i) \mid i \in [1, D] \wedge \hat{y}_i = y_i\}$ # Filter rationales using ground truth answers
  $\mathcal{D}_n^{\text{rat}} \leftarrow \{(x_i, \hat{r}_i^{\text{rat}}, y_i) \mid i \in [1, D] \wedge \hat{y}_i \neq y_i \wedge \hat{y}_i^{\text{rat}} = y_i\}$ # Filter rationalized rationales
  $M_n \leftarrow \text{train}(M, \mathcal{D}_n \cup \mathcal{D}_n^{\text{rat}})$ # Finetune the original model on correct solutions - inner loop
end for

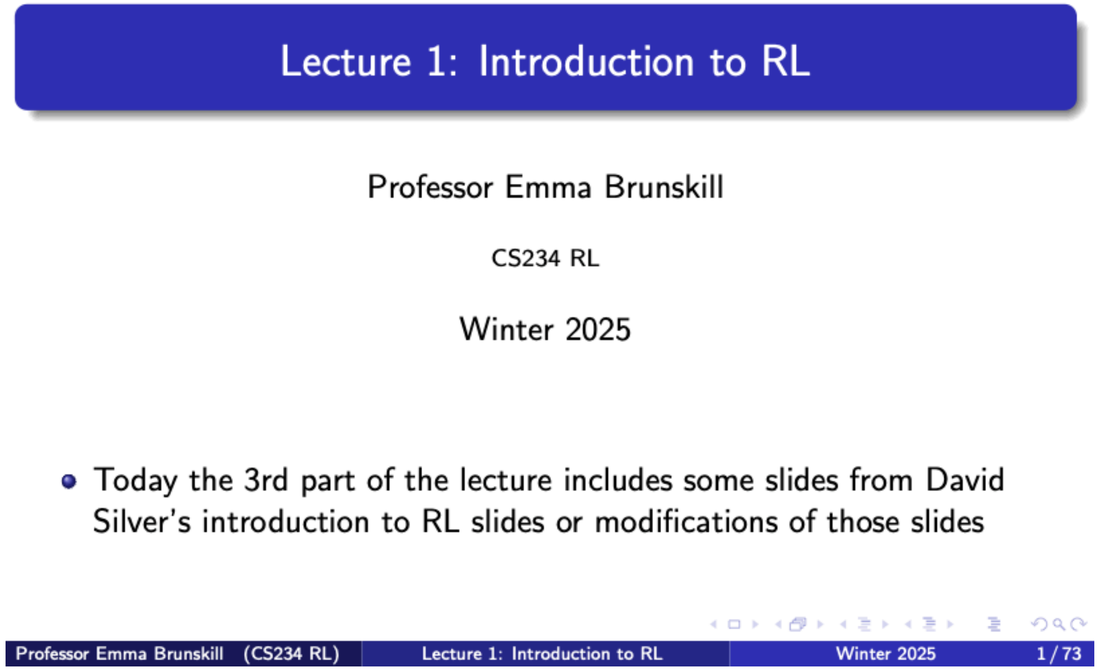
Wasting incorrect solutions :(

	GSM8K Test Accuracy (%)	Train Data Used (%)
Few-shot Direct GPT-J	3.0	~0
Few-shot CoT GPT-J	3.1	~0
GPT-J Direct Finetuned	5.8	100
STaR without rationalization	10.1	25.0
STaR with rationalization	**10.7**	28.7

Reinforcement Learning

Sources

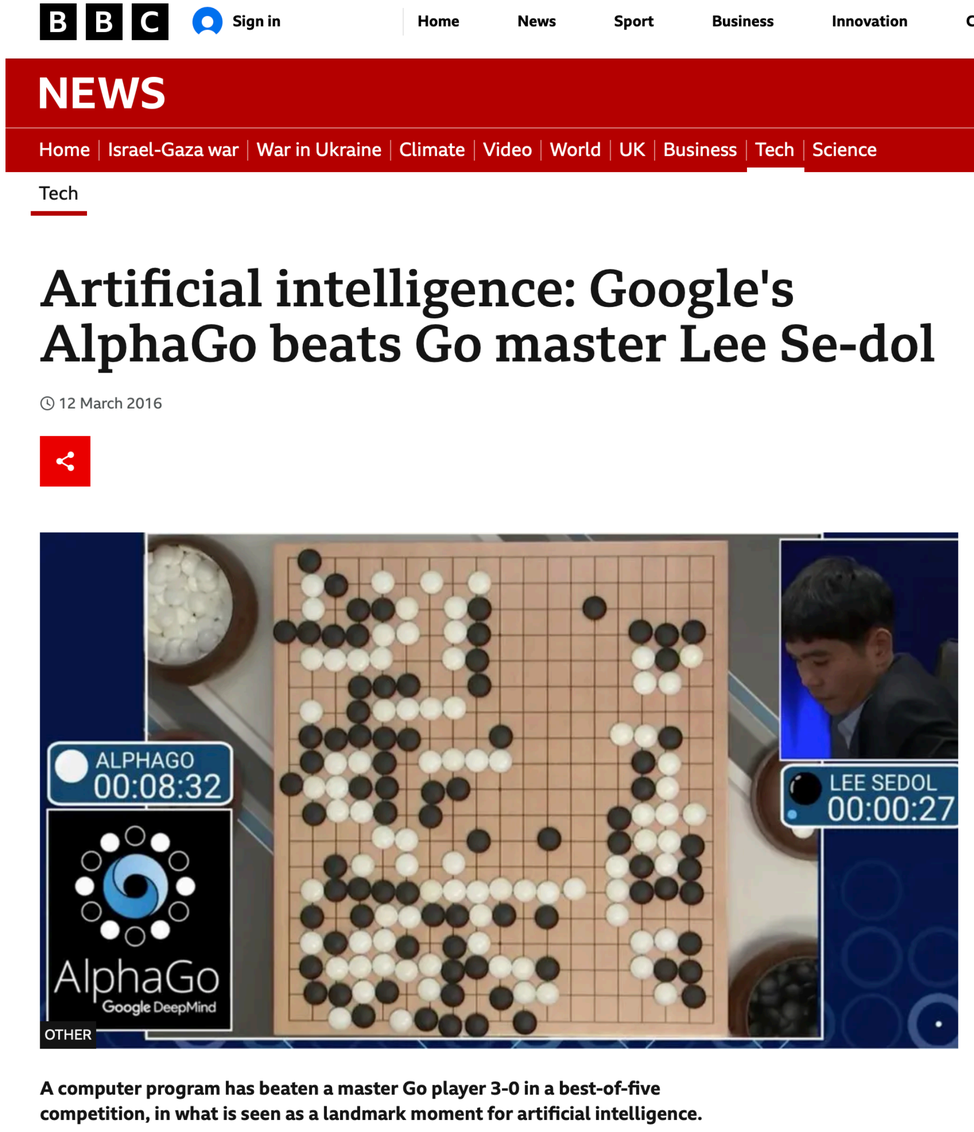


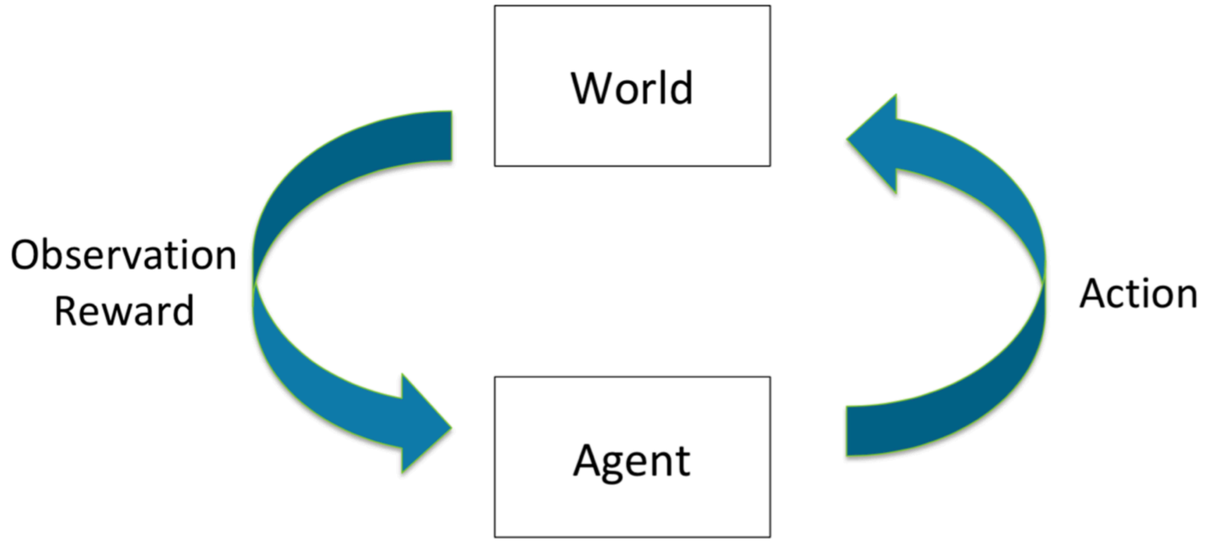


- With many slides from or derived from David Silver and John Schulman and Pieter Abbeel

Reinforcement learning (RL)

- Reinforcement learning: Learning through experience/data to make good decisions under uncertainty.
- Builds strongly from theory and ideas starting in the 1950s with Richard Bellman.
- Commonly used for robotics.
- Impressive success cases:
  - math, physics (nuclear fusion), games, finance, …





Goal: Select actions to maximize total expected future reward

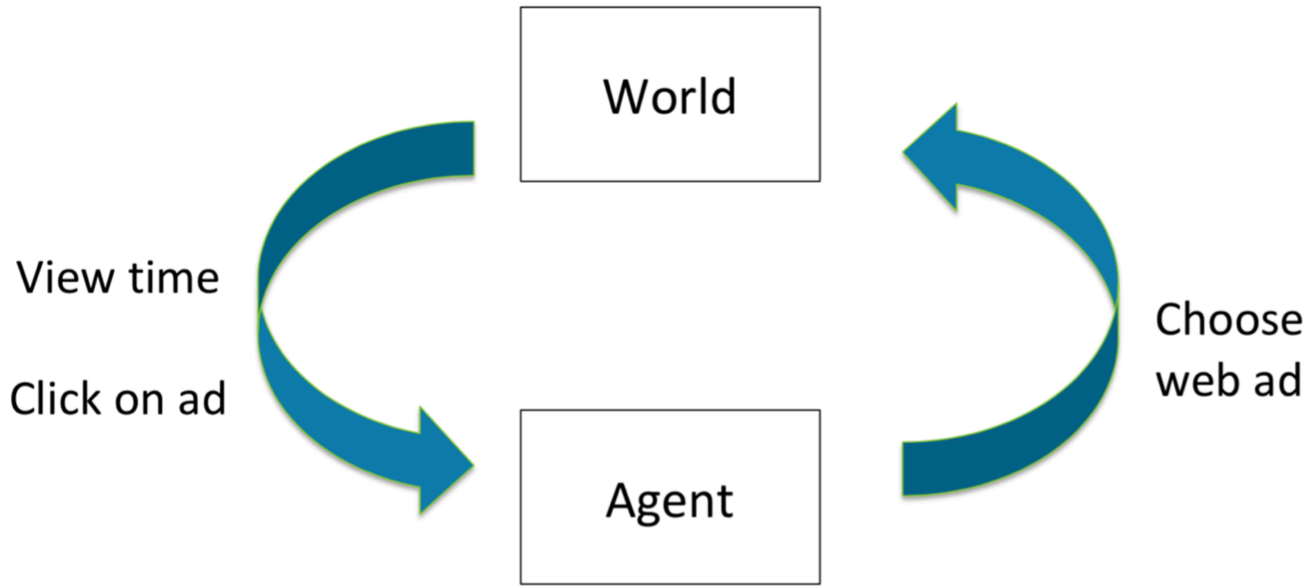
May require balancing immediate & long term rewards

RL properties

- Delayed consequences
  - Decisions now can impact things much later.
  - E.g., Saving for retirement, actions in a game, or actions of a robot.
- Challenges:
  - When planning: decisions involve reasoning about not just immediate benefit of a decision but also its longer term consequences.
  - When learning: temporal credit assignment is hard (what caused later high or low rewards?)

RL properties

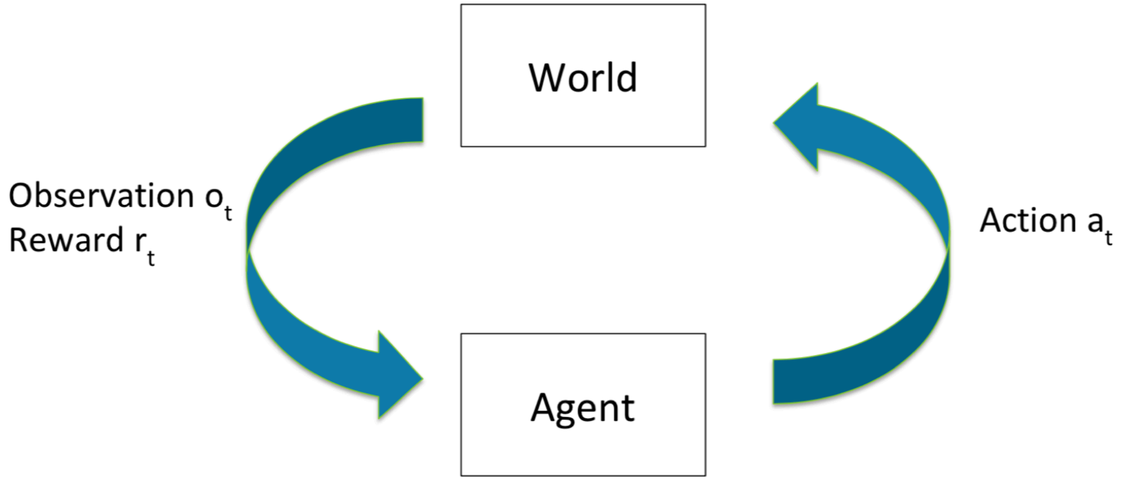
Web advertising



RL properties

- RL: Learning about the world by making decisions
  - Learn to ride a bike by trying (and failing)
- We want a mapping from current state to the action
  - However, the space of states is usually very large
    - E.g., in (video) games or robotics
    - A simple video game frame, may contain 200x300 resolution, and each pixel can have 256^3 values.

RL properties



- History $h_t = (a_1, o_1, r_1, \ldots, a_t, o_t, r_t)$
- State is information assumed to determine what happens next Function of history: $s_t = (h_t)$
- Markov assumption
  - Information state: sufficient statistic of the history
  - $\mathbb{P}(s_{t+1} \mid s_t, a_t) = \mathbb{P}(s_{t+1} \mid h_t, a_t)$ (next state might not be deterministic)
  - In practice we may assume that $o_t = s_t$

Markov Decision Process (MDP)

- S is a set of Markov states.
- We have a set of actions A.
- We have a transition function specifying $\mathbb{P}(s_{t+1} = s' \mid s_t = s, a_t = a)$.
- We have reward function $R(s_t = s, a_t = a) = \mathbb{E}[r_t \mid s_t = s, a_t = a]$.
- Discount factor $\gamma \in [0,1]$.
  - Total reward is computed by $r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots$
  - Preferring earlier rewards (like humans)
  - If horizon is finite, we can use $\gamma = 1$.

Markov Decision Process (MDP)

- Policy: $\pi(a \mid s) = \mathbb{P}(a_t = a \mid s_t = s)$

$$R^{\pi}(s) = \sum_{a \in A} \pi(a|s) R(s, a)$$

$$P^{\pi}(s'|s) = \sum_{a \in A} \pi(a|s) P(s'|s, a)$$

$$V^{\pi}(s) = \underbrace{R^{\pi}(s)}_{\text{Immediate reward}} + \underbrace{\gamma \sum_{s' \in S} P^{\pi}(s'|s) V^{\pi}(s')}_{\text{Discounted sum of future rewards}}$$

$$V^{\pi} = R^{\pi} + \gamma P^{\pi} V^{\pi}$$

Markov Decision Process (MDP)

Policy: $\pi(a \mid s) = \mathbb{P}(a_t = a \mid s_t = s)$

State-action value: $Q^{\pi}(s, a) = R(s, a) + \gamma \sum_{s'} P(s' \mid s, a) V^{\pi}(s')$

How to find the optimal policy? (Assuming finite states)

- Policy iteration
  - Compute values for a policy and use it to get a better policy.
- Value iteration
  - Compute optimal value for horizon k and then $k + 1$.

RL for LMs

Agent (policy) = language model

Initial state

Problem: Suppose a and b are positive real numbers with $a > b$ and $ab = 8$. Find the minimum value of $\frac{a^2+b^2}{a-b}$.

Ground truth solution: We can write $\frac{a^2+b^2}{a-b} = \frac{a^2+b^2-2ab+16}{a-b} = \frac{(a-b)^2+16}{a-b} = a - b + \frac{16}{a-b}$. By AM-GM, $a - b + \frac{16}{a-b} \geq 2\sqrt{(a-b) \cdot \frac{16}{a-b}} = 8$. Equality occurs when $a - b = 4$ and $ab = 8$. We can solve these equations to find $a = 2\sqrt{3} + 2$ and $b = 2\sqrt{3} - 2$. Thus, the minimum value is $\boxed{8}$.

Steps = actions

reward = 1 if answer is correct

Intuition: Sparse-reward MDP with deterministic dynamics

LMs and RL

Policy is parametrized: $\pi_\theta(s, a) = \mathbb{P}(a \mid s; \theta)$

Goal: maximize $V(\theta) := V(s_0, \theta)$

We need to compute the policy gradient $\nabla_\theta V(s_0, \theta)$

- assuming π_θ is differentiable

$V(\theta) = \sum_\tau \mathbb{P}(\tau; \theta) R(\tau)$ where τ presents a trajectory $(s_0, a_0, r_0, \ldots)$

Policy Gradient

$$
\begin{aligned}
\nabla_\theta V(\theta) &= \nabla_\theta \sum_\tau P(\tau;\theta) R(\tau) \\
&= \sum_\tau \nabla_\theta P(\tau;\theta) R(\tau) \\
&= \sum_\tau \frac{P(\tau;\theta)}{P(\tau;\theta)} \nabla_\theta P(\tau;\theta) R(\tau) \\
&= \sum_\tau P(\tau;\theta) R(\tau) \underbrace{\frac{\nabla_\theta P(\tau;\theta)}{P(\tau;\theta)}}_{\text{likelihood ratio}} \\
&= \sum P(\tau;\theta) R(\tau) \nabla_\theta \log P(\tau;\theta)
\end{aligned}
$$

Approximation using m samples under π_θ:

$$
\nabla_\theta V(\theta) \approx \hat{g} = (1/m) \sum_{i=1}^{m} R(\tau^{(i)}) \nabla_\theta \log P(\tau^{(i)};\theta)
$$

Policy Gradient

Approximation using m samples under π_θ:

$$\nabla_\theta V(\theta) \approx \hat{g} = (1/m) \sum_{i=1}^{m} R(\tau^{(i)}) \nabla_\theta \log P(\tau^{(i)}; \theta)$$

$$\begin{aligned}
\nabla_\theta \log P(\tau^{(i)}; \theta) &= \nabla_\theta \log \left[\underbrace{\mu(s_0)}_{\text{Initial state distrib.}} \prod_{t=0}^{T-1} \underbrace{\pi_\theta(a_t|s_t)}_{\text{policy}} \underbrace{P(s_{t+1}|s_t, a_t)}_{\text{dynamics model}} \right] \\
&= \nabla_\theta \left[\log \mu(s_0) + \sum_{t=0}^{T-1} \log \pi_\theta(a_t|s_t) + \log P(s_{t+1}|s_t, a_t) \right] \\
&= \sum_{t=0}^{T-1} \underbrace{\nabla_\theta \log \pi_\theta(a_t|s_t)}_{\text{no dynamics model required!}} \qquad \textit{score function}
\end{aligned}$$

REINFORCE:

Initialize policy parameters θ arbitrarily

for each episode $\{s_1, a_1, r_2, \cdots, s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ **do**

  for $t = 1$ to $T - 1$ **do**

    $\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) G_t$

  endfor

endfor

return θ

where $G_t = \sum_{k=t}^{T-1} \gamma^{k-t} R_{k+1}$ is the return.

Generating more data

STaR

STaR: Self-Taught Reasoner
Bootstrapping Reasoning With Reasoning

Eric Zelikman[*1], **Yuhuai Wu**[*12], **Jesse Mu**[1], **Noah D. Goodman**[1]
[1]Department of Computer Science, Stanford University
[2] Google Research
{ezelikman, yuhuai, muj, ngoodman}@stanford.edu

Algorithm 1 STaR

Input M: a pretrained LLM; dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{D}$ (w/ few-shot prompts)

$M_0 \leftarrow M$ # Copy the original model
for n **in** $1...N$ **do** # Outer loop
  $(\hat{r}_i, \hat{y}_i) \leftarrow M_{n-1}(x_i) \quad \forall i \in [1, D]$ # Perform rationale generation
  $(\hat{r}_i^{\text{rat}}, \hat{y}_i^{\text{rat}}) \leftarrow M_{n-1}(\text{add\_hint}(x_i, y_i)) \quad \forall i \in [1, D]$ # Perform rationalization
  $\mathcal{D}_n \leftarrow \{(x_i, \hat{r}_i, y_i) \mid i \in [1, D] \wedge \hat{y}_i = y_i\}$ # Filter rationales using ground truth answers
  $\mathcal{D}_n^{\text{rat}} \leftarrow \{(x_i, \hat{r}_i^{\text{rat}}, y_i) \mid i \in [1, D] \wedge \hat{y}_i \neq y_i \wedge \hat{y}_i^{\text{rat}} = y_i\}$ # Filter rationalized rationales
  $M_n \leftarrow \text{train}(M, \mathcal{D}_n \cup \mathcal{D}_n^{\text{rat}})$ # Finetune the original model on correct solutions - inner loop
end for

STaR is very similar to REINFORCE

$$J(M, X, Y) = \sum_i \mathbb{E}_{\hat{r}_i, \hat{y}_i \sim p_M(\cdot|x_i)} \mathbb{1}(\hat{y}_i = y_i), \tag{1}$$

$$\nabla J(M, X, Y) = \sum_i \mathbb{E}_{\hat{r}_i, \hat{y}_i \sim p_M(\cdot|x_i)} \left[\mathbb{1}(\hat{y}_i = y_i) \cdot \nabla \log p_M(\hat{y}_i, \hat{r}_i \mid x_i)\right], \tag{2}$$

Policy Gradient with baseline

We can deduct a baseline value from the return:

$$\nabla_\theta \mathbb{E}_\tau[R] = \mathbb{E}_\tau \left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t|s_t;\theta) \left(\sum_{t'=t}^{T-1} r_{t'} - b(s_t) \right) \right] \qquad \text{(wlog } \gamma = 1\text{)}$$

+ The estimator remains unbiased.

+ We can reduce the variance.

- In particular the expected return or the state-value $V^\pi(s)$ is a great baseline.
  - We'd like to assess how much the return is better/worse than the average.
  - We call $G_t - b(s_t) = G_t - V^\pi(s_t) = A^\pi(s_t, a_t)$ the advantage.

Challenges of policy gradient

- Distance in parameter space vs. distance in policy space
  - Choosing the right learning rate can be hard. —> stability issues.
- Sample efficiency is poor.
  - PG is on-policy => we need to get rid of the data after each gradient step.
  - Ideally we'd like to use old data for multiple gradient steps.
- Solution: We can use samples from the old policy, as long as:
  - Policies are close, in particular, low KL divergence.
  - Importance sampling weight is included.

Challenges of policy gradient

- Assuming $D_{KL}(\pi' \| \pi)$ is small, we can equivalently find π' maximizing:

$$\mathop{\mathrm{E}}_{\tau \sim \pi}\left[\sum_{t=0}^{\infty} \gamma^t \frac{\pi'(a_t|s_t)}{\pi(a_t|s_t)} A^{\pi}(s_t, a_t)\right]$$

where π is the old policy.

stability challenge

Proximal Policy Optimization (PPO)

Proximal Policy Optimization Algorithms

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov
OpenAI
{joschu, filip, prafulla, alec, oleg}@openai.com

- Using KL penalty: $\theta_{k+1} = \arg\max_{\theta} \mathcal{L}_{\theta_k}(\theta) - \beta_k \bar{D}_{KL}(\theta||\theta_k)$

- Using clipped objective: $\mathcal{L}^{CLIP}_{\theta_k}(\theta) = \mathop{\mathrm{E}}_{\tau\sim\pi_k}\left[\sum_{t=0}^{T}\left[\min(r_t(\theta)\hat{A}^{\pi_k}_t, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}^{\pi_k}_t)\right]\right]$

  where $r_t(\theta) = \pi_\theta(a_t \mid s_t)/\pi_{\theta_k}(a_t \mid s_t)$.

Proximal Policy Optimization (PPO)

Algorithm PPO with Clipped Objective

Input: initial policy parameters θ_0, clipping threshold ϵ
for $k = 0, 1, 2, \ldots$ **do**
Collect set of partial trajectories $\mathcal{D}_k$ on policy $\pi_k = \pi(\theta_k)$
Estimate advantages $\hat{A}_t^{\pi_k}$ using any advantage estimation algorithm
Compute policy update

$$\theta_{k+1} = \arg\max_{\theta} \mathcal{L}_{\theta_k}^{CLIP}(\theta)$$

by taking K steps of minibatch SGD (via Adam), where

$$\mathcal{L}_{\theta_k}^{CLIP}(\theta) = \mathop{\mathrm{E}}_{\tau \sim \pi_k} \left[\sum_{t=0}^{T} \left[\min(r_t(\theta)\hat{A}_t^{\pi_k}, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t^{\pi_k}) \right] \right]$$

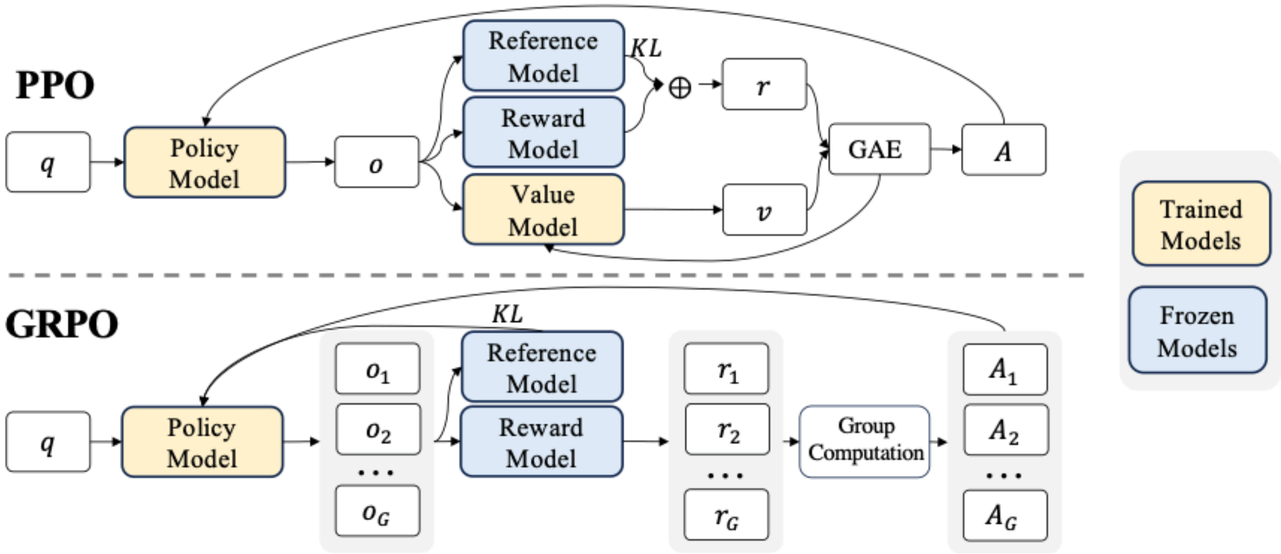
end for

Advantage is often estimated by learning a separate value model, e.g.,

$$A_t = G_t - V_\phi(s_t)$$

Group Relative Policy Optimization (GRPO)

- GRPO generates multiple trajectories per question, and uses the averaged (normalized) reward as advantage. —> no need for a value model!



DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models

Zhihong Shao[1,2*†], Peiyi Wang[1,3*†], Qihao Zhu[1,3*†], Runxin Xu[1], Junxiao Song[1]
Xiao Bi[1], Haowei Zhang[1], Mingchuan Zhang[1], Y.K. Li[1], Y. Wu[1], Daya Guo[1*]

$$A_i = \frac{r_i - mean(r)}{std(r)}$$

GRPO is now commonly used for training reasoning models.

Sparse rewards problem

- Hard questions may not provide any reward.
- Solutions:
  - Using a curriculum: training on simpler questions first.
  - Helping the model by making the question easier.
  - Process Reward Model (PRM): rewarding individual steps of the solution.