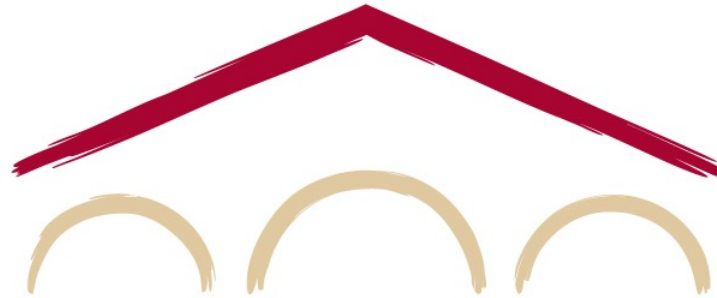


# Natural Language Processing with Deep Learning

## CS224N/Ling284



Shikhar Murty

Lecture 14: Reasoning and Agents

# Lecture Plan

## Lecture 14: Reasoning and Agents

1. Reasoning in Language Models [35 mins]
  2. Mini-break [5 mins]
  3. Language Model Agents [40 mins]
- **Announcements**
    - Project Milestone due on Wed May 22<sup>nd</sup> at 4:30 pm
    - Your Project Mentors have already reached out to you (If not, let us know via Ed!)
    - Guest lectures on May 21<sup>st</sup> and May 28<sup>th</sup> : Students get 0.75% per guest lecture for attending live or writing a reaction paragraph (More details will be on Ed)

# Reasoning (with Large Language Models)

# What is Reasoning?

Using *facts* and *logic* to arrive at an answer



# What is Reasoning?

Using *facts* and *logic* to arrive at an answer

**Deductive Reasoning:** Use logic to go from premise to firm conclusion

Premise: All mammals have kidneys  
Premise: All whales are mammals  
Conclusion: All whales have kidneys

# What is Reasoning?

Using *facts* and *logic* to arrive at an answer

**Deductive Reasoning:** Use logic to go from premise to firm conclusion

Premise: All mammals have kidneys  
Premise: All whales are mammals  
Conclusion: All whales have kidneys

**Inductive Reasoning:** From observation, predict a likely conclusion

Observation: When we see a creature with wings, it is usually a bird  
Observation: We see a creature with wings.  
Conclusion: The creature is likely to be a bird

# What is Reasoning?

Using *facts* and *logic* to arrive at an answer

**Deductive Reasoning:** Use logic to go from premise to firm conclusion

Premise: All mammals have kidneys  
Premise: All whales are mammals  
Conclusion: All whales have kidneys

**Inductive Reasoning:** From observation, predict a likely conclusion

Observation: When we see a creature with wings, it is usually a bird  
Observation: We see a creature with wings.  
Conclusion: The creature is likely to be a bird

**Abductive Reasoning:** From observation, predict the most likely explanation

Observation: The car cannot start and there is a puddle of liquid under the engine.  
Likely Explanation: The car has a leak in the radiator

# Reasoning: Formal vs Informal

**Formal Reasoning:** Follows formal rules of logic along with axiomatic knowledge to derive conclusions.

**Informal Reasoning:** Uses intuition, experience, common sense to arrive at answers.

For most of this lecture, by “reasoning” we mean informal deductive reasoning, often involving multiple steps

# Reasoning in Large Language Models

Large Language models are **REALLY GOOD** at predicting **plausible continuations of text**, that respect **constraints in the input**, and align **well with human preferences**.

**Question:** Can **current** LLMs reason?

# Reasoning in Large Language Models: prompting

## Chain-of-thought prompting:

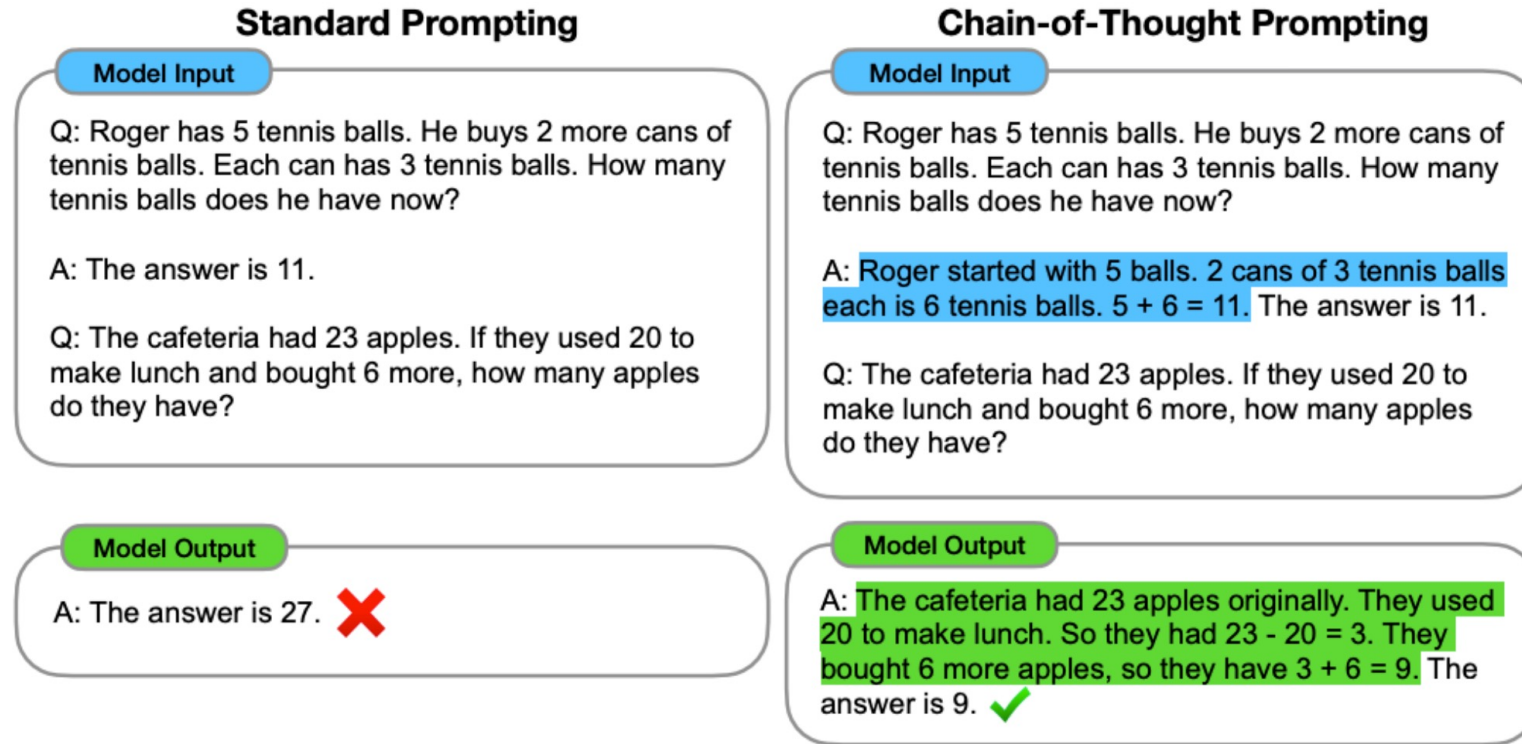


Figure 1: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

# Reasoning in Large Language Models: prompting

## Zero-shot CoT prompting:

(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. X*

(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 X*

(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. ✓*

(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?

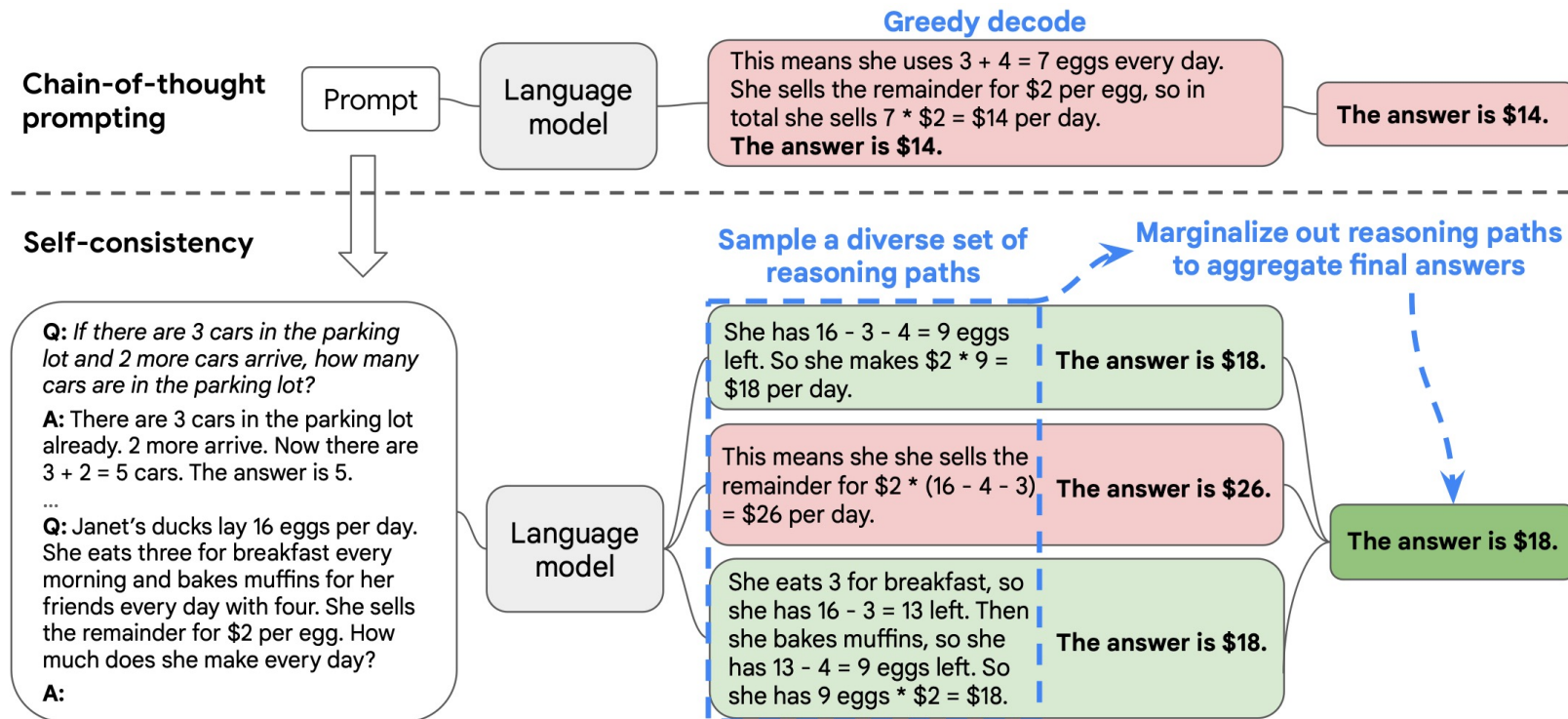
A: **Let's think step by step.**

(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. ✓*

# Reasoning in Large Language Models: prompting

CoT with "Self-consistency": Replace greedy decoding with an ensemble of samples...

**Main idea:** correct reasoning processes have greater agreement than incorrect processes.





# Reasoning in Large Language Models: prompting

	Method	AddSub	MultiArith	ASDiv	AQuA	SVAMP	GSM8K
	Previous SoTA	<b>94.9<sup>a</sup></b>	60.5 <sup>a</sup>	75.3 <sup>b</sup>	37.9 <sup>c</sup>	57.4 <sup>d</sup>	35 <sup>e</sup> / 55 <sup>g</sup>
UL2-20B	CoT-prompting	18.2	10.7	16.9	23.6	12.6	4.1
	Self-consistency	24.8 (+6.6)	15.0 (+4.3)	21.5 (+4.6)	26.9 (+3.3)	19.4 (+6.8)	7.3 (+3.2)
LaMDA-137B	CoT-prompting	52.9	51.8	49.0	17.7	38.9	17.1
	Self-consistency	63.5 (+10.6)	75.7 (+23.9)	58.2 (+9.2)	26.8 (+9.1)	53.3 (+14.4)	27.7 (+10.6)
PaLM-540B	CoT-prompting	91.9	94.7	74.0	35.8	79.0	56.5
	Self-consistency	93.7 (+1.8)	99.3 (+4.6)	81.9 (+7.9)	48.3 (+12.5)	86.6 (+7.6)	74.4 (+17.9)

**Out-performs regular CoT on a variety of benchmarks**

# Reasoning in Large Language Models: prompting

	Method	AddSub	MultiArith	ASDiv	AQuA	SVAMP	GSM8K
	Previous SoTA	<b>94.9<sup>a</sup></b>	60.5 <sup>a</sup>	75.3 <sup>b</sup>	37.9 <sup>c</sup>	57.4 <sup>d</sup>	35 <sup>e</sup> / 55 <sup>g</sup>
UL2-20B	CoT-prompting	18.2	10.7	16.9	23.6	12.6	4.1
	Self-consistency	24.8 (+6.6)	15.0 (+4.3)	21.5 (+4.6)	26.9 (+3.3)	19.4 (+6.8)	7.3 (+3.2)
LaMDA-137B	CoT-prompting	52.9	51.8	49.0	17.7	38.9	17.1
	Self-consistency	63.5 (+10.6)	75.7 (+23.9)	58.2 (+9.2)	26.8 (+9.1)	53.3 (+14.4)	27.7 (+10.6)
PaLM-540B	CoT-prompting	91.9	94.7	74.0	35.8	79.0	56.5
	Self-consistency	93.7 (+1.8)	99.3 (+4.6)	81.9 (+7.9)	48.3 (+12.5)	86.6 (+7.6)	74.4 (+17.9)

Out-performs regular CoT on a variety of benchmarks

	GSM8K	MultiArith	SVAMP	ARC-e	ARC-c
CoT (Wei et al., 2022)	17.1	51.8	38.9	75.3	55.1
Ensemble (3 sets of prompts)	18.6 ± 0.5	57.1 ± 0.7	42.1 ± 0.6	76.6 ± 0.1	57.0 ± 0.2
Ensemble (40 prompt permutations)	19.2 ± 0.1	60.9 ± 0.2	42.7 ± 0.1	76.9 ± 0.1	57.0 ± 0.1
Self-Consistency (40 sampled paths)	<b>27.7 ± 0.2</b>	<b>75.7 ± 0.3</b>	<b>53.3 ± 0.2</b>	<b>79.3 ± 0.3</b>	<b>59.8 ± 0.2</b>

Self-consistency is doing more than simple ensembling

# Reasoning in Large Language Models: prompting

## Problem decomposition with Least-to-Most prompting

### Stage 1: Decompose Question into Subquestions

**Q:** It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The water slide closes in 15 minutes. How many times can she slide before it closes?

Language Model

**A:** To solve "How many times can she slide before it closes?", we need to first solve: "How long does each trip take?"

# Reasoning in Large Language Models: prompting

## Problem decomposition with Least-to-Most prompting

### Stage 1: Decompose Question into Subquestions

Q: It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The water slide closes in 15 minutes. How many times can she slide before it closes?

Language Model

A: To solve "How many times can she slide before it closes?", we need to first solve: "How long does each trip take?"

### Stage 2: Sequentially Solve Subquestions

Subquestion 1

It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.

Q: How long does each trip take?

Language Model

A: It takes Amy 4 minutes to climb and 1 minute to slide down.  $4 + 1 = 5$ . So each trip takes 5 minutes.

Append model answer to Subquestion 1

It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.

Q: How long does each trip take?

A: It takes Amy 4 minutes to climb and 1 minute to slide down.  $4 + 1 = 5$ . So each trip takes 5 minutes.

Language Model

A: The water slide closes in 15 minutes. Each trip takes 5 minutes. So Amy can slide  $15 \div 5 = 3$  times before it closes.

Subquestion 2

Q: How many times can she slide before it closes?

# Reasoning in Large Language Models: prompting

## Least-to-Most prompting for Math reasoning

Q: Elsa has 5 apples. Anna has 2 more apples than Elsa. How many apples do they have together?

A: Anna has 2 more apples than Elsa, so Anna has  $2 + 5 = 7$  apples. Elsa and Anna have  $5 + 7 = 12$  apples together. The answer is 12.

**Standard CoT prompting**

Q: Elsa has 5 apples. Anna has 2 more apples than Elsa. How many apples do they have together?

A: Let's break down this problem: 1. How many apples does Anna have? 2. How many apples do Elsa and Anna have together?

1. Anna has 2 more apples than Elsa. So Anna has  $2 + 5 = 7$  apples.

2. Elsa and Anna have  $5 + 7 = 12$  apples together.

**Least-to-most prompting**

Q: {question}

A: Let's break down this problem:

—  
The answer is:

# Reasoning in Large Language Models: prompting

## Least-to-Most prompting for Math reasoning

Accuracy by Steps (GSM8K)	All	2 Steps	3 Steps	4 steps	$\geq 5$ steps
Least-to-Most	<b>62.39</b>	74.53	<b>68.91</b>	<b>59.73</b>	<b>45.23</b>
Chain-of-Thought	60.87	<b>76.68</b>	67.29	59.39	39.07

Generalizes to more #steps than in-context example!

# Reasoning in Large Language Models: prompting

## Least-to-Most prompting for Math reasoning

Accuracy by Steps (GSM8K)	All	2 Steps	3 Steps	4 steps	$\geq 5$ steps
Least-to-Most	<b>62.39</b>	74.53	<b>68.91</b>	<b>59.73</b>	<b>45.23</b>
Chain-of-Thought	60.87	<b>76.68</b>	67.29	59.39	39.07

Generalizes to more #steps than in-context example!

Prompting method	Accuracy
Zero-Shot	16.38
Standard prompting	17.06 <sup>3</sup>
Chain-of-Thought (original)	61.18
Chain-of-Thought (1-shot)	60.88
Least-to-Most (1-shot)	62.39
Chain-of-Thought (best)	<b>68.61<sup>3</sup></b>
Least-to-Most (best)	68.01

But with enough prompt engineering, CoT  $\approx$  Least-to-Most

# Reasoning in ~~Large~~ Language Models via distillation

So far, we've only looked at prompting >100B parameter models for reasoning

Can we get reasoning-like behavior with smaller LMs by teaching them to imitate larger models?



# ORCA: Instruction-tuning small LMs with CoT Rationales

1. Collect a wide variety of instructions from the FLAN-v2 collection

Mixture Name	Sampling Algorithm	Original Size	Sampled
CoT	No Sampling	150K	150K
NIV2	Stratified Sampling	5M	440K
FLAN2021	Stratified Sampling	>28.9M	2.5M
T0	Stratified Sampling	85.7M	2M
Dialog	Skipped	22.5M	0

Table 3: Construction of our training data with 5 million samples.

# ORCA: Instruction-tuning small LMs with CoT Rationales

1. Collect a wide variety of instructions from the FLAN-v2 collection.
2. Prompt GPT4 or ChatGPT with these instructions along with a **system message**

**System Instruction:** You are an AI assistant. User will give you a task. Your goal is to complete the task as faithfully as you can. While performing the task think step-by-step and justify your steps.

**User Instruction:** Use the given data to calculate the median.

**Input:** [7, 3, 8, 2, 10]

**Output:** To calculate the median, I will follow these steps:

Step 1: Arrange the data in ascending order [2, 3, 7, 8, 10]

Step 2: Determine if the data set has an odd or even number of values

The data set has 5 values, which is an odd number.

Step 3: Find the middle value

Since the data set has an odd number of values, the median is the middle value in the ordered list. The middle value in the list [2, 3, 7, 8, 10] is 7.

# ORCA: Instruction-tuning small LMs with CoT Rationales

1. Collect a wide variety of instructions from the FLAN-v2 collection.
2. Prompt GPT4 or ChatGPT with these instructions along with a **system message**
3. Finetune Llama-13b on outputs generated via ChatGPT + GPT4

# ORCA: Instruction-tuning small LMs with CoT Rationales

Evaluation: *BigBench-hard*, a collection of 23 Big-Bench tasks with a special focus on multi-step reasoning

# ORCA: Instruction-tuning small LMs with CoT Rationales

Evaluation: **BigBench-hard**, a collection of 23 Big-Bench tasks with a special focus on multi-step reasoning

## Boolean Expressions

**Q:** True and False and not True and True is

**A:** Let's think step by step.

Remember that (i) expressions inside brackets are always evaluated first and that (ii) the order of operations from highest priority to lowest priority is "not", "and", "or", respectively.

We first simplify this expression "Z" as follows: "Z = True and False and not True and True = A and B" where "A = True and False" and "B = not True and True".

Let's evaluate A: A = True and False = False.

Let's evaluate B: B = not True and True = not (True and True) = not (True) = False.

Plugging in A and B, we get: Z = A and B = False and False = False. So the answer is False.

# ORCA: Instruction-tuning small LMs with CoT Rationales

Evaluation: **BigBench-hard**, a collection of 23 Big-Bench tasks with a special focus on multi-step reasoning

## Data Understanding

**Q:** Tomorrow is 11/12/2019. What is the date one year ago from today in MM/DD/YYYY?

**Options:**

- (A) 09/04/2018
- (B) 11/11/2018
- (C) 08/25/2018
- (D) 11/02/2018
- (E) 11/04/2018

**A:** Let's think step by step.

If tomorrow is 11/12/2019, then today is 11/11/2019. The date one year ago from today is 11/11/2018. So the answer is **(B)**.

# ORCA: Instruction-tuning small LMs with CoT Rationales

Evaluation: **BigBench-hard**, a collection of 23 Big-Bench tasks with a special focus on multi-step reasoning

## Geometric Shapes

**Q:** This SVG path element `<path d="M 14.19,26.04 L 51.43,39.21 L 58.44,36.69 L 56.63,30.17 L 48.53,26.66 L 14.19,26.04"/>` draws a

**Options:**

- (A) circle
- (B) heptagon
- (C) hexagon
- (D) kite
- (E) line
- (F) octagon
- (G) pentagon
- (H) rectangle
- (I) sector
- (J) triangle

**A:** Let's think step by step.

This SVG path element contains "M" and "L" commands. M takes two parameters (x,y) and moves the current point to the coordinates (x,y). L takes two parameters (x,y) and draws a line from the previous coordinate to the new coordinate (x,y).

This path can be decomposed into 6 separate commands.

- (1) M 14.19,26.04: Move the current point to 14.19,26.04.
- (2) L 51.43,39.21: Create a line from 14.19,26.04 to 51.43,39.21.
- (3) L 58.44,36.69: Create a line from 51.43,39.21 to 58.44,36.69.
- (4) L 56.63,30.17: Create a line from 58.44,36.69 to 56.63,30.17.
- (5) L 48.53,26.66: Create a line from 56.63,30.17 to 48.53,26.66.
- (6) L 14.19,26.04: Create a line from 48.53,26.66 to 14.19,26.04.

This SVG path starts at point 14.19,26.04, creates five consecutive and touching lines, and then returns back its starting point, thereby creating a five-sided shape. It does not have any curves or arches. "pentagon" is the only five-sided polygon on the list. So the answer is **(G)**.

# ORCA: Instruction-tuning small LMs with CoT Rationales

Task	ChatGPT	GPT-4	Vicuna-13B	Orca-13B
Boolean Expressions	82.8	77.6	40.8	<b>72.0</b> (76.5%)
Causal Judgement	57.2	59.9	42.2	<b>59.9</b> (41.8%)
Date Understanding	42.8	74.8	10.0	<b>50.0</b> (400.0%)
Disambiguation QA	57.2	69.2	18.4	<b>63.6</b> (245.7%)
Formal Fallacies	53.6	64.4	47.2	<b>56.0</b> (18.6%)
Geometric Shapes	25.6	40.8	3.6	<b>20.8</b> (477.8%)
Hyperbaton	69.2	62.8	44.0	<b>64.0</b> (45.5%)
Logical Deduction (5 objects)	38.8	66.8	4.8	<b>39.6</b> (725.0%)
Logical Deduction (7 objects)	39.6	66.0	1.2	<b>36.0</b> (2900.0%)
Logical Deduction (3 objects)	60.4	94.0	16.8	<b>57.6</b> (242.9%)
Movie Recommendation	55.4	79.5	43.4	<b>78.3</b> (80.6%)
Navigate	55.6	68.8	46.4	<b>57.6</b> (24.1%)
Penguins in a Table	45.9	76.7	15.1	<b>42.5</b> (181.8%)
Reasoning about Colored Objects	47.6	84.8	12.0	<b>48.4</b> (303.3%)
Ruin Names	56.0	89.1	15.7	<b>39.5</b> (151.2%)
Salient Translation Error Detection	40.8	62.4	2.0	<b>40.8</b> (1940.0%)
Snarks	59.0	87.6	28.1	<b>62.4</b> (122.0%)
Sports Understanding	79.6	84.4	48.4	<b>67.2</b> (38.8%)
Temporal Sequences	35.6	98.0	16.0	<b>72.0</b> (350.0%)
Tracking Shuffled Objects (5 objects)	18.4	25.2	9.2	<b>15.6</b> (69.6%)
Tracking Shuffled Objects (7 objects)	15.2	25.2	5.6	<b>14.0</b> (150.0%)
Tracking Shuffled Objects (3 objects)	31.6	42.4	23.2	<b>34.8</b> (50.0%)
Web of Lies	56.0	49.6	41.2	<b>51.2</b> (24.3%)
Average	48.9	67.4	23.3	<b>49.7</b> (113.7%)

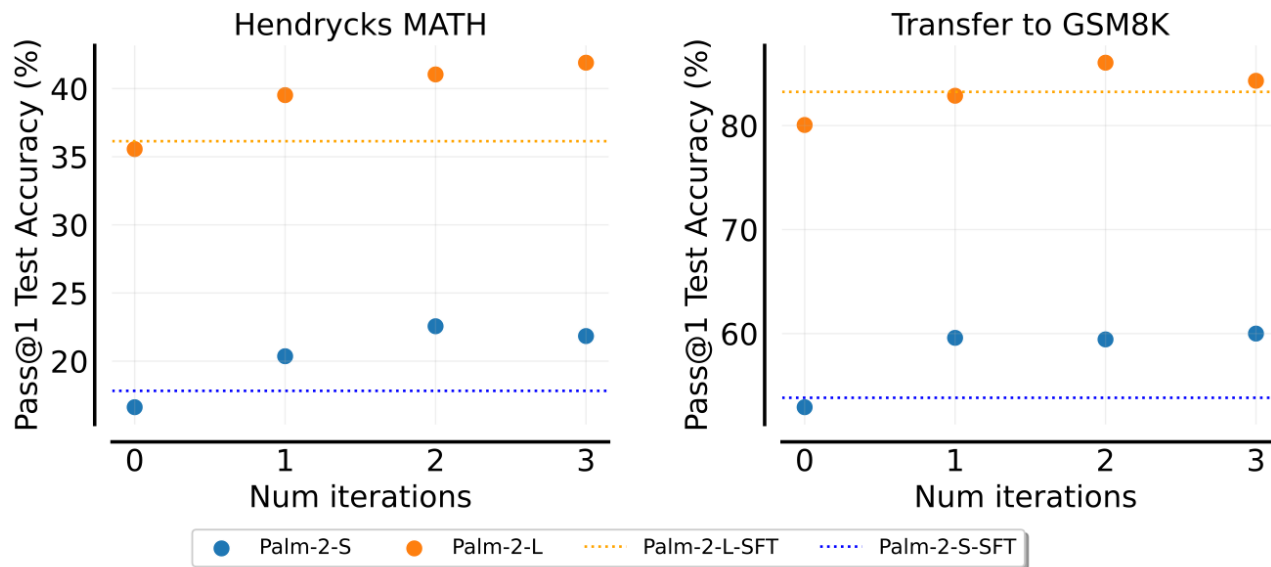
- Outperforms Vicuna-13B
- Outperforms ChatGPT!
- GPT-4 has potential data contamination issues with Bigbench-hard



# Reasoning by Finetuning LMs on their own outputs?

ReSTEM alternates between the following two steps:

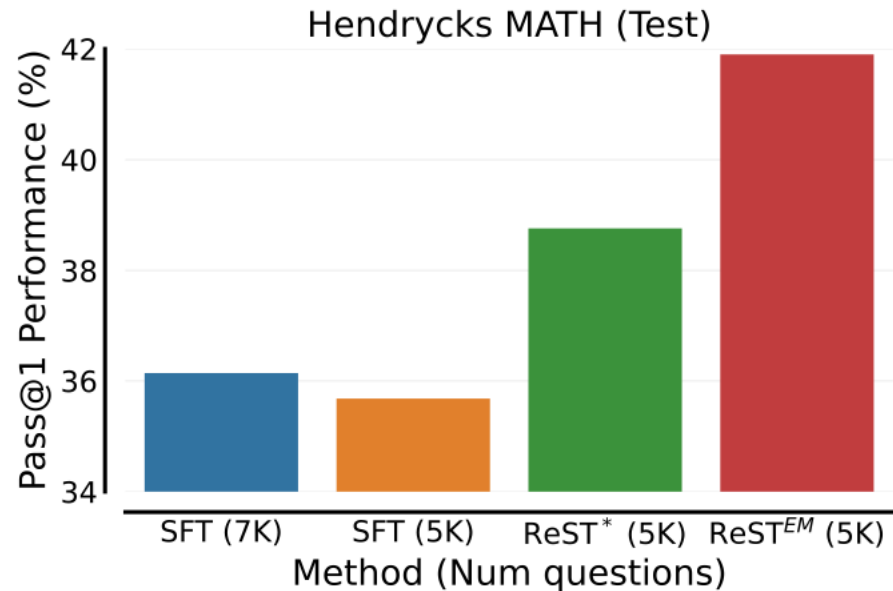
1. Generate (E-Step): Given reasoning problem, sample multiple solutions from language model. Filter based on some (problem specific) function [answer correctness for math problems]
2. Improve (M-Step): Update the language model to maximize probability of filtered solutions, using supervised finetuning



# Reasoning by Finetuning LMs on their own outputs?

ReST<sup>EM</sup> alternates between the following two steps:

1. Generate (E-Step): Given reasoning problem, sample multiple solutions from language model. Filter based on some (problem specific) function [answer correctness for math problems]
2. Improve (M-Step): Update the language model to maximize probability of filtered solutions, using supervised finetuning



# Can Language Models Reason?

The  
Economist

Science and technology | Generative AI

Large language models' ability to generate text also lets them plan and reason

TechTalks

Large language models have a reasoning problem



The New York Times

*Microsoft Says New A.I. Shows Signs of Human Reasoning*

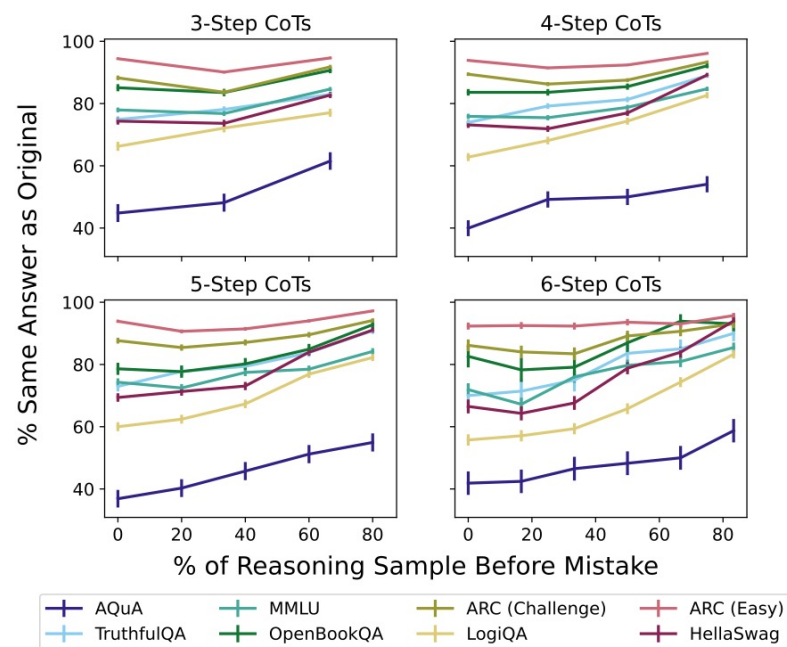
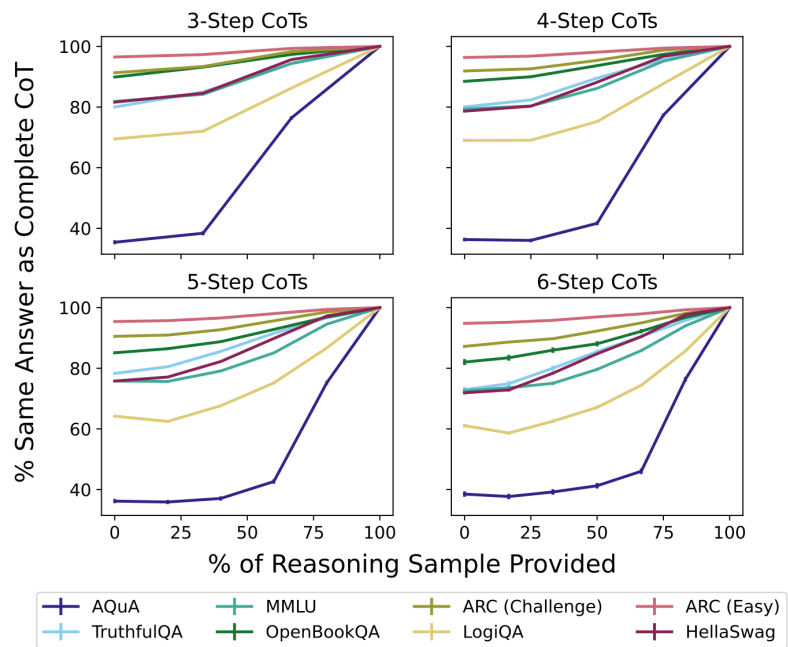


WIRED

Some Glimpse AGI in ChatGPT. Others Call It a Mirage

# Can Language Models Reason?

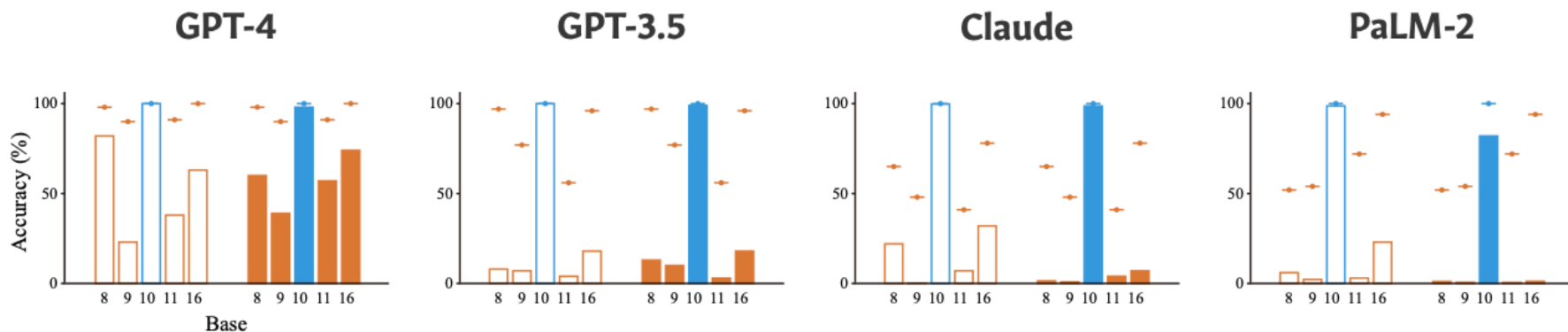
## CoT Rationales are often not faithful



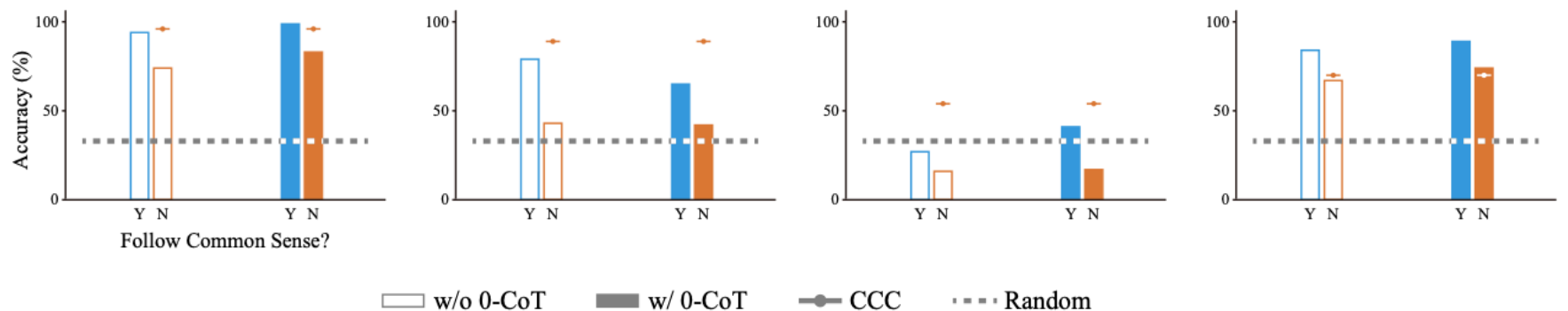
# Can Language Models Reason?

## Reasoning vs Memorization: Using Counterfactuals

**Arithmetic**  
Two-digit addition



**Logic**  
First-order logic deduction in natural language



□ w/o 0-CoT    ■ w/ 0-CoT    ● CCC    - - - Random

# Can Language Models Reason?

## Reasoning vs Memorization: Using Counterfactuals

### Original transformation types

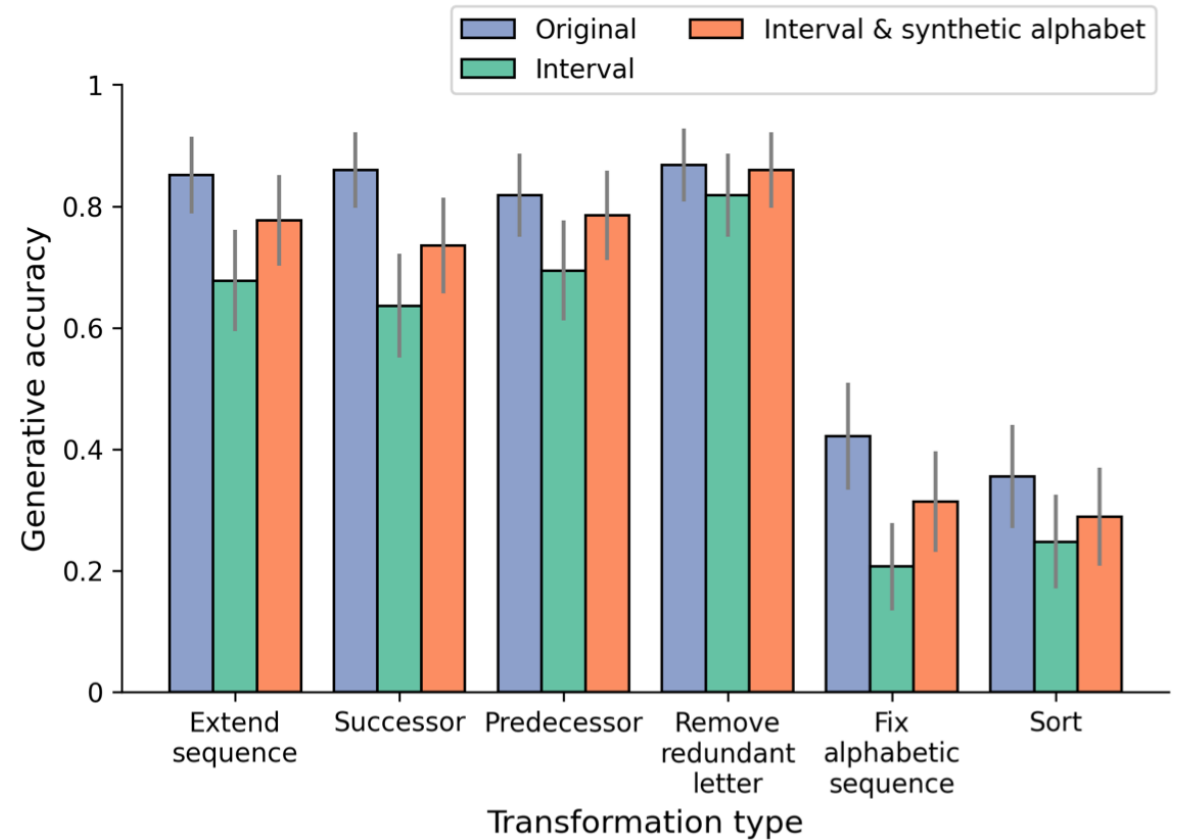
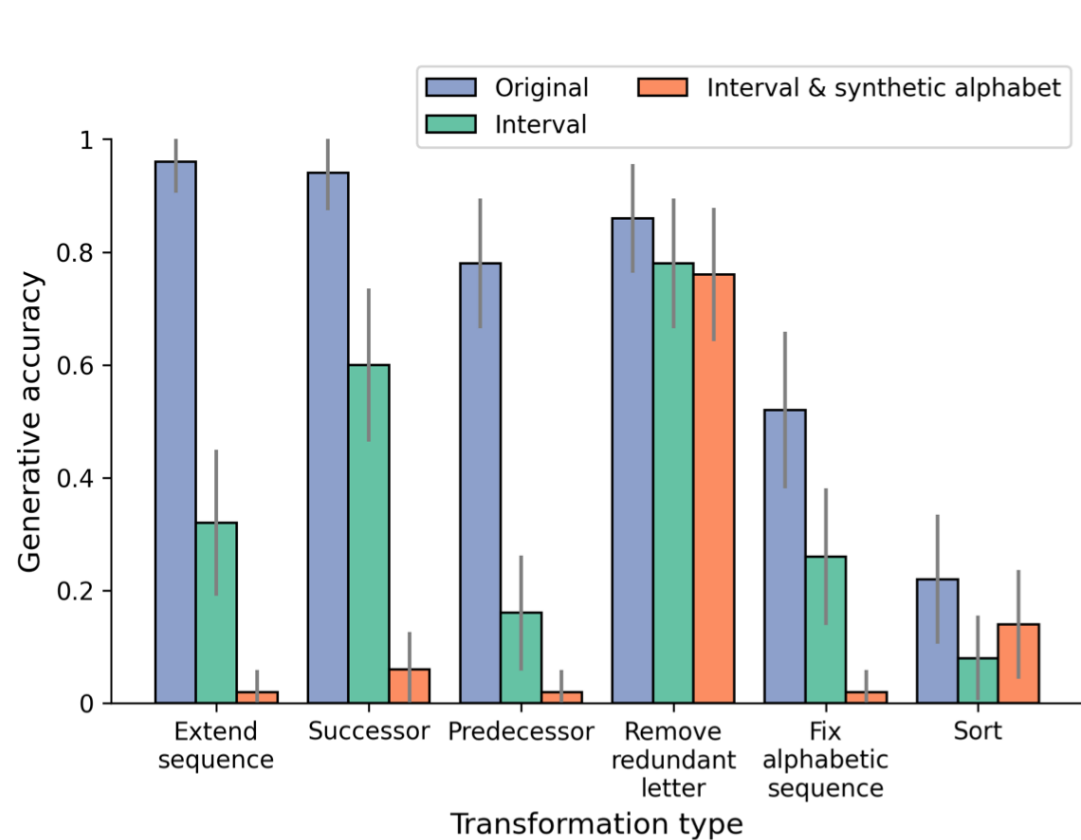
Extend sequence	Successor	Predecessor
a b c d → a b c d e	a b c d → a b c e	b c d e → a c d e
i j k l → i j k l m	i j k l → i j k m	i j k l → h j k l
Remove redundant letter	Fix alphabetic sequence	Sort
a b b c d e → a b c d e	a b c w e → a b c d e	a d c b e → a b c d e
i j k k l m → i j k l m	i j k x m → i j k l m	k j m l i → i j k l m

### Modified transformation types

Extend sequence	Successor	Predecessor
a b c d → a b c d f	a b c d → a b c f	c d e f → a d e f
i j k l → i j k l n	i j k l → i j k n	j k l m → h k l m
Remove redundant letter	Fix alphabetic sequence	Sort
a c e g i i → a c e g i	a c e g o → a c e g i	k f a p u → a f k p u
i k k m o q → i k m o q	i k x o q → i k m o q	i m k o q → i k m o q

# Can Language Models Reason?

## Reasoning vs Memorization: Using Counterfactuals

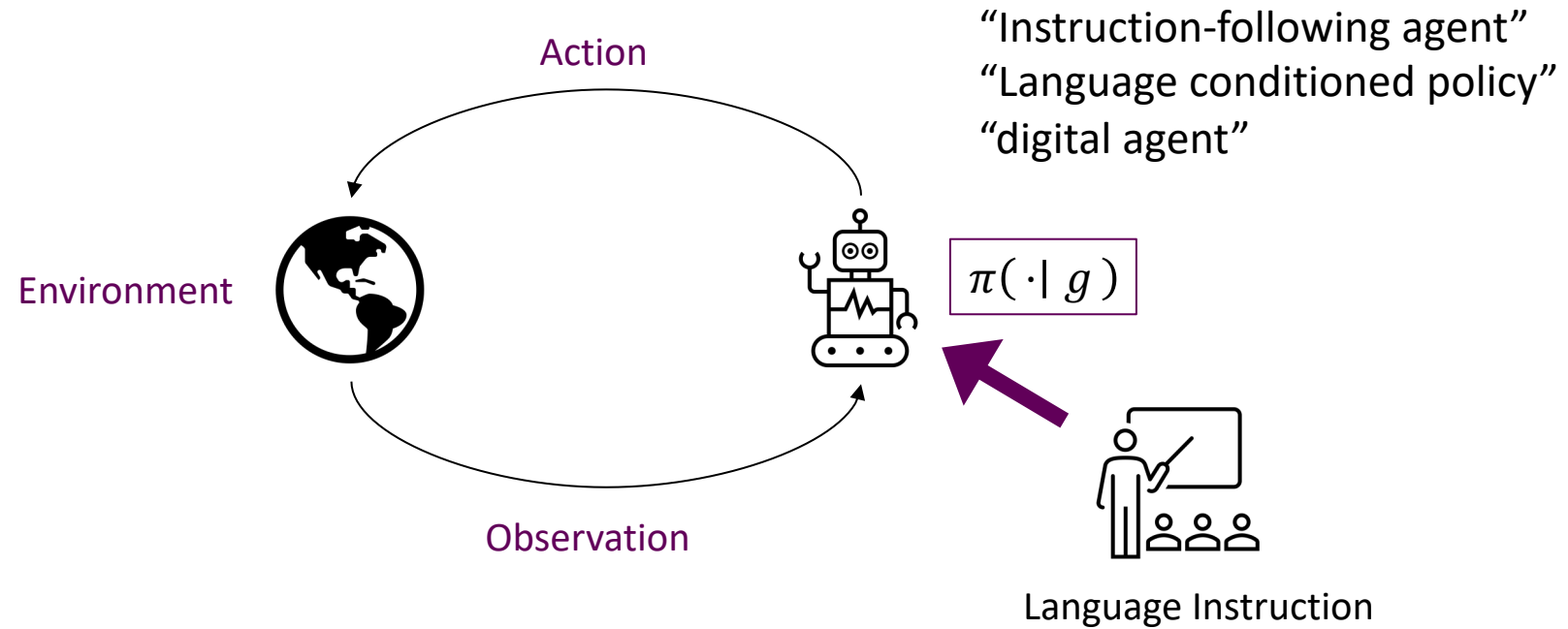


# Language Model Agents

with some slides borrowed from Frank Xu (CMU)

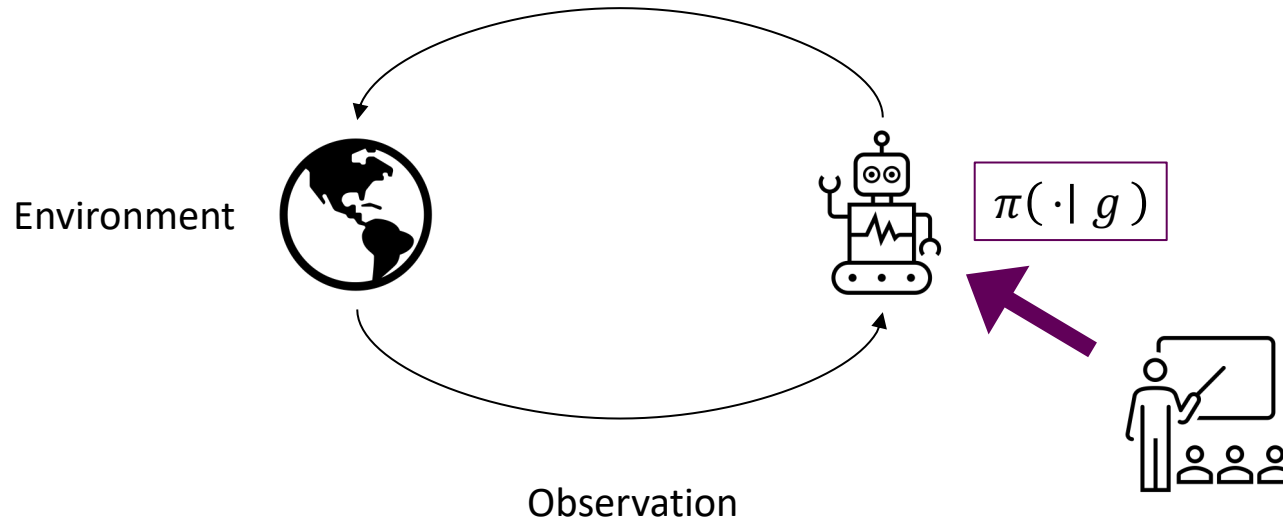


# Some Terminology

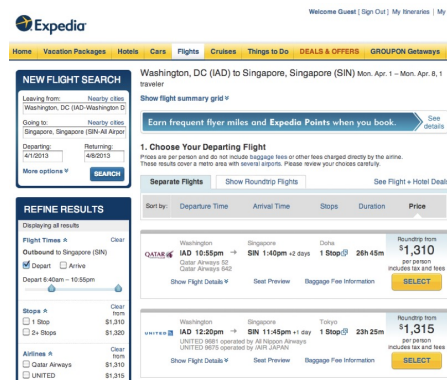


# Some Terminology

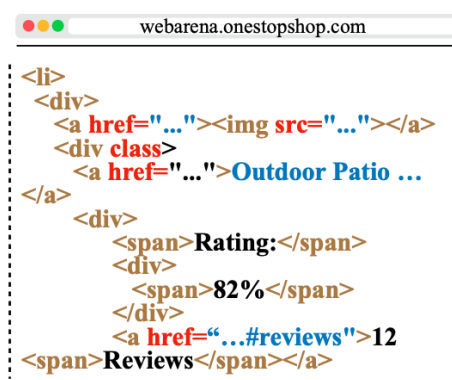
Type ... on ..., Click on ..., Choose ... from dropdown, ...



“Book a flight from San Francisco to New York”

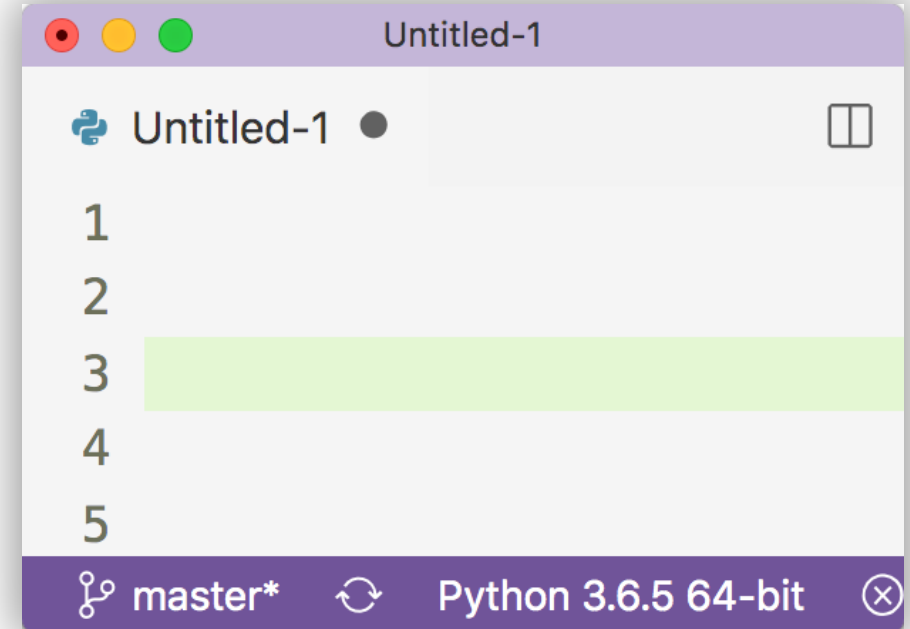
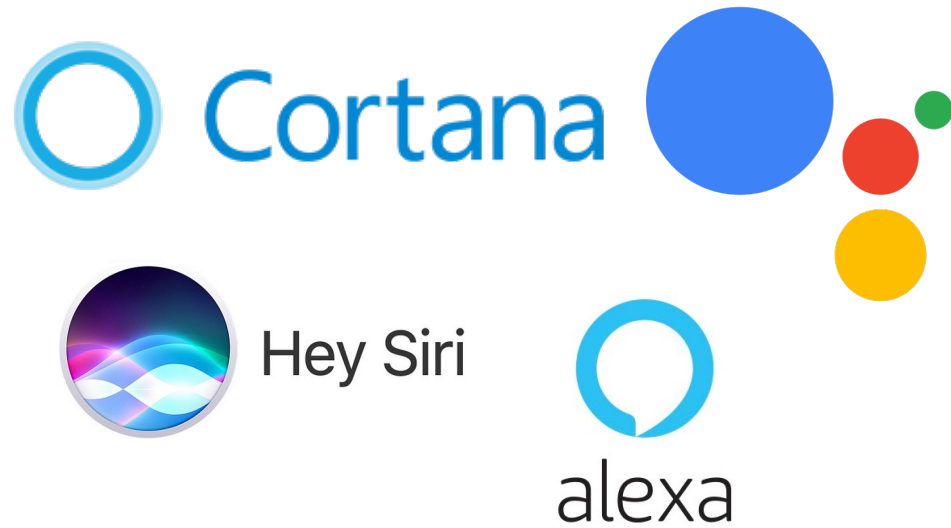


Raw pixels as observation?






HTML DOM as observation?




# Applications: Natural Language Interfaces




## Virtual Assistants

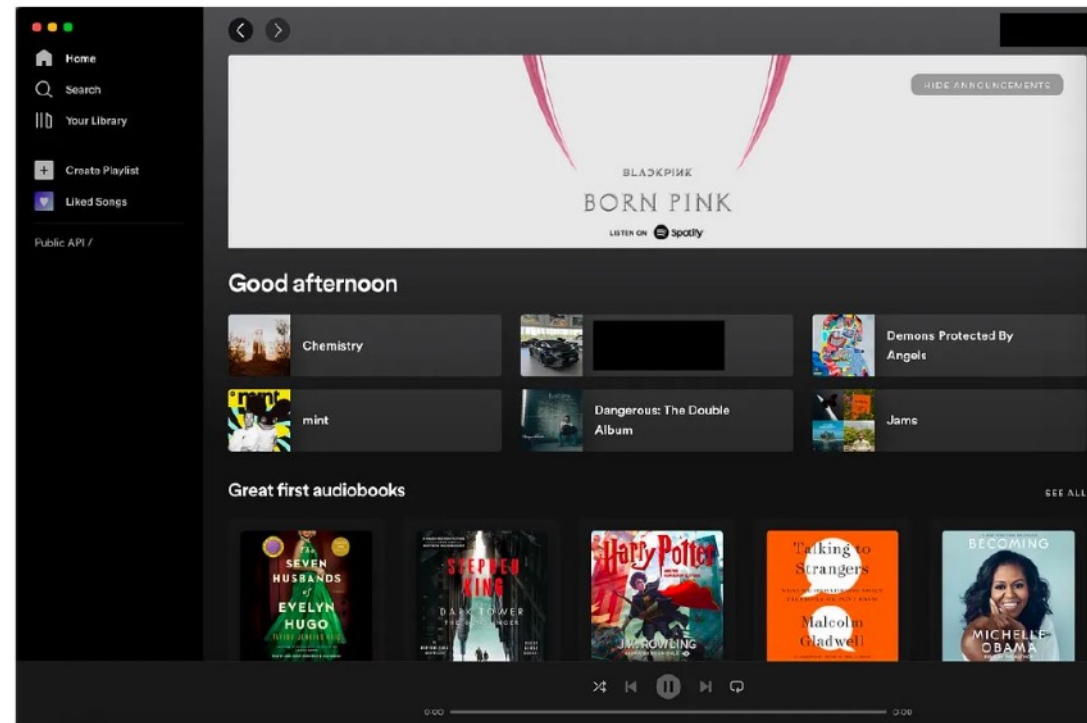
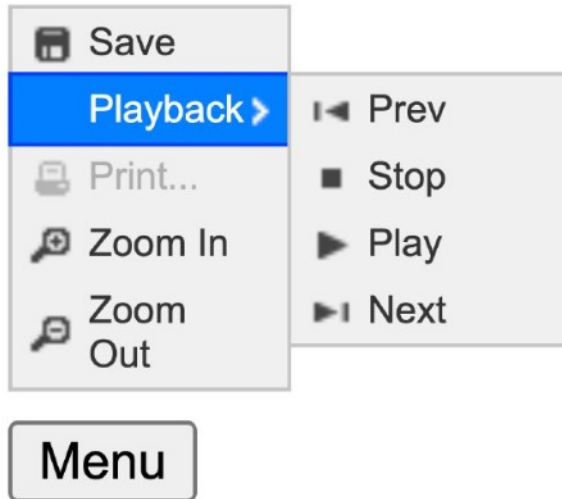
-  *Set an alarm at 7 AM*
-  *Remind me for the meeting at 5pm*
-  *Play Jay Chou's latest album*

## Natural Language Programming

-  *Sort my\_list in descending order*
-  *Copy my\_file to home folder*
-  *Dump my\_dict as a csv file output.csv*

# Applications: UI automation

Click the "Menu" button, and then find and click on the item with the  icon.














“Play some synthwave songs”

# Applications: Multi-step “Tool use”

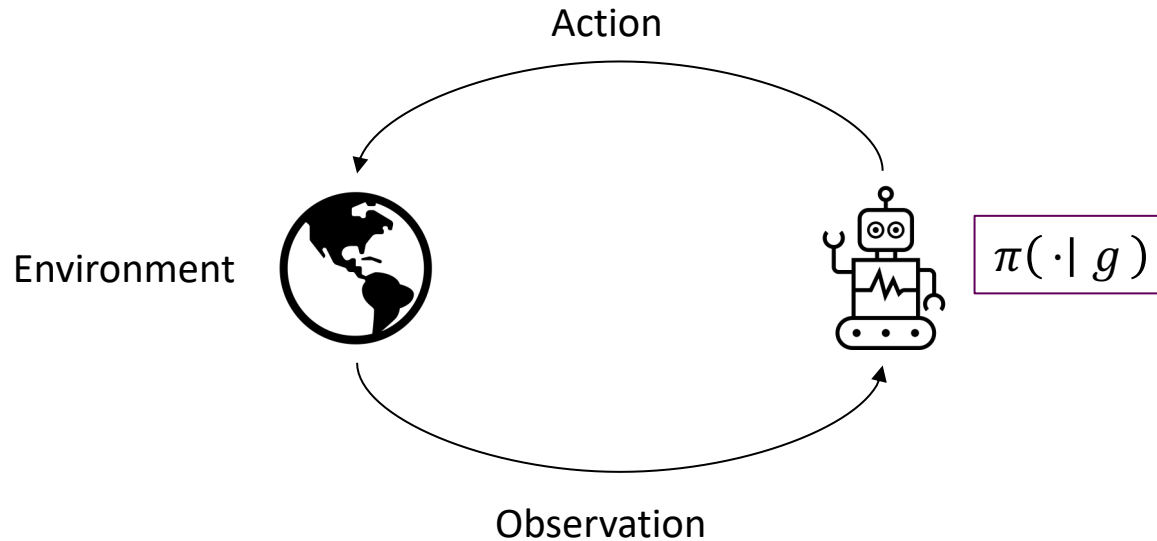
## ChatGPT plugins

We've implemented initial support for plugins in ChatGPT. Plugins allow the language model to help ChatGPT access external data, perform computations, or

[ChatGPT plugins](#)

 <p><b>Expedia</b> Bring your trip plans to life—get there, stay there, find things to see and do.</p>	 <p><b>FiscalNote</b> Provides and enables access to select market-leading, real-time data sets for legal, political, and regulatory data and information.</p>	 <p><b>Instacart</b> Order from your favorite local grocery stores.</p>	 <p><b>KAYAK</b> Search for flights, stays and rental cars. Get recommendations for all the places you can go within your budget.</p>
 <p><b>Klarna Shopping</b> Search and compare prices from thousands of online shops.</p>	 <p><b>Milo Family AI</b> Giving parents superpowers to turn the manic to magic, 20 minutes each day. Ask: Hey Milo, what's magic today?</p>	 <p><b>OpenTable</b> Provides restaurant recommendations, with a direct link to book.</p>	 <p><b>Shop</b> Search for millions of products from the world's greatest brands.</p>
 <p><b>Speak</b> Learn how to say anything in another language with Speak, your AI-powered language tutor.</p>	 <p><b>Wolfram</b> Access computation, math, curated knowledge &amp; real-time data through Wolfram Alpha and Wolfram Language.</p>	 <p><b>Zapier</b> Interact with over 5,000+ apps like Google Sheets, Trello, Gmail, HubSpot, Salesforce, and more.</p>	

# Instruction following agents [Pre LLMs]

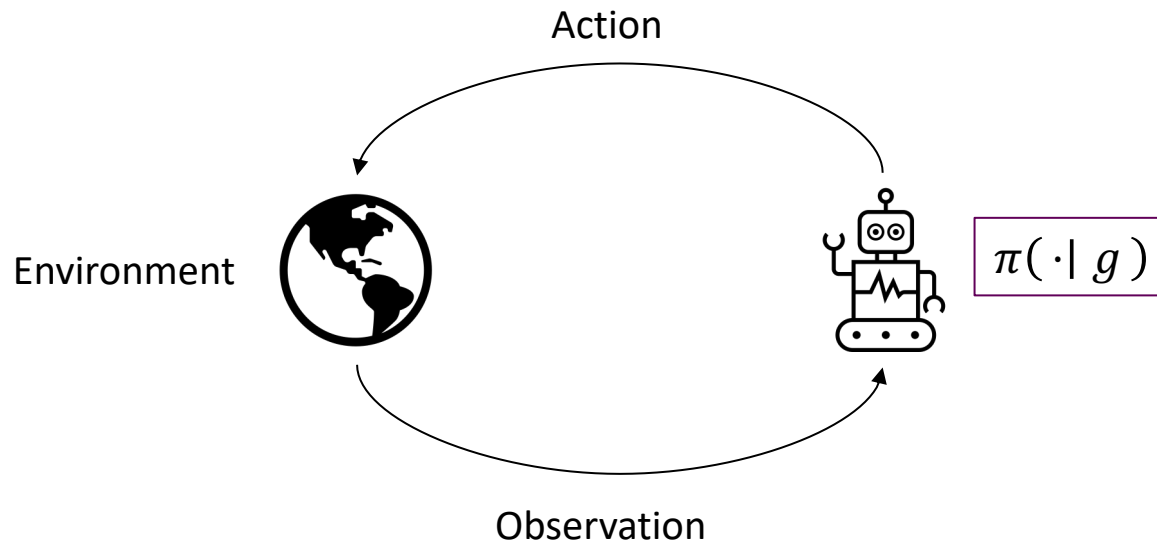


- a) What states border Texas  
 $\lambda x.state(x) \wedge borders(x, texas)$
- b) What is the largest state  
 $\arg \max(\lambda x.state(x), \lambda x.size(x))$
- c) What states border the state that borders the most states  
 $\lambda x.state(x) \wedge borders(x, \arg \max(\lambda y.state(y), \lambda y.count(\lambda z.state(z) \wedge borders(y, z))))$

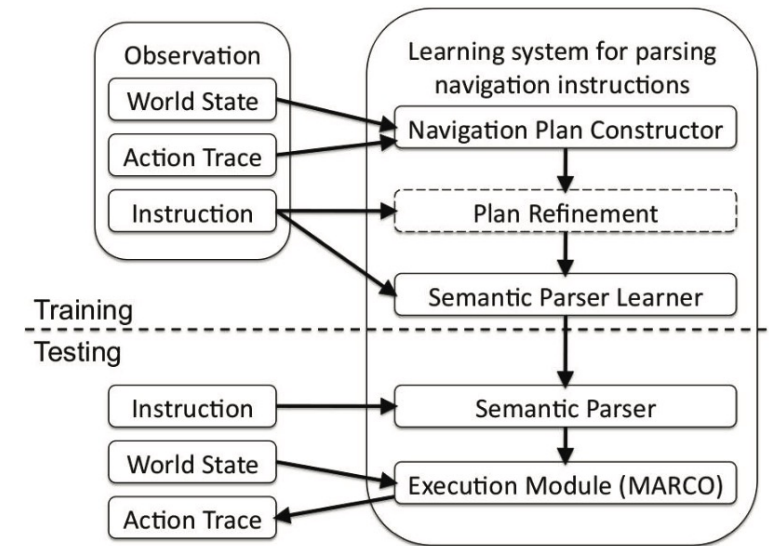
Idea #1: Directly map from instructions to action sequences like Machine Translation  
[works well for simple grounded environments like text2sql, knowledge graph querying]

$$\max_{\theta} p_{\theta}(\{a_1, a_2, \dots\} | g)$$

# Instruction following agents [Pre LLMs]



Idea #2: Infer **executable**, latent plans from (instruction, trajectory) pairs and train a model to go from instructions to plans



**Instruction:** “Place your back against the wall of the ‘T’ intersection. Turn left. Go forward along the pink-flowered carpet hall two segments to the intersection with the brick hall. This intersection contains a hatrack. Turn left. Go forward three segments to an intersection with a bare concrete hall, passing a lamp. This is Position 5.”

**Parse:** Turn ( ),  
 Verify ( back: WALL ),  
 Turn ( LEFT ),  
 Travel ( ),  
 Verify ( side: BRICK HALLWAY ),  
 Turn ( LEFT ),  
 Travel ( steps: 3 ),  
 Verify ( side: CONCRETE HALLWAY )

# Instruction following agents [Pre LLMs]

*u:* click **Run**, and press **OK** after typing **secpol.msc** in the **open** box.

*a:* *c:* left-click *R:* [ **Run...** ]

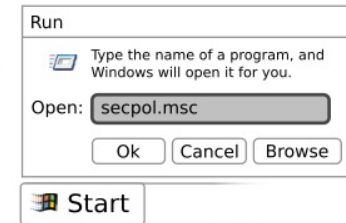
*ε:*



*u:* click **Run**, and press **OK** after typing **secpol.msc** in the **open** box.

*a:* left-click **Run...** *c:* type-into *R:* [ **open** "secpol.msc" ]

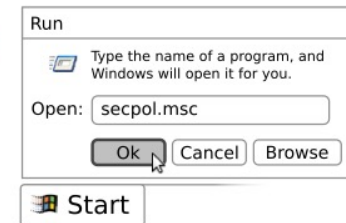
*ε:*



*u:* click **Run**, and press **OK** after typing **secpol.msc** in the **open** box.

*a:* left-click **Run...** type-into **open** "secpol.msc" *c:* left-click *R:* [ **OK** ]

*ε:*

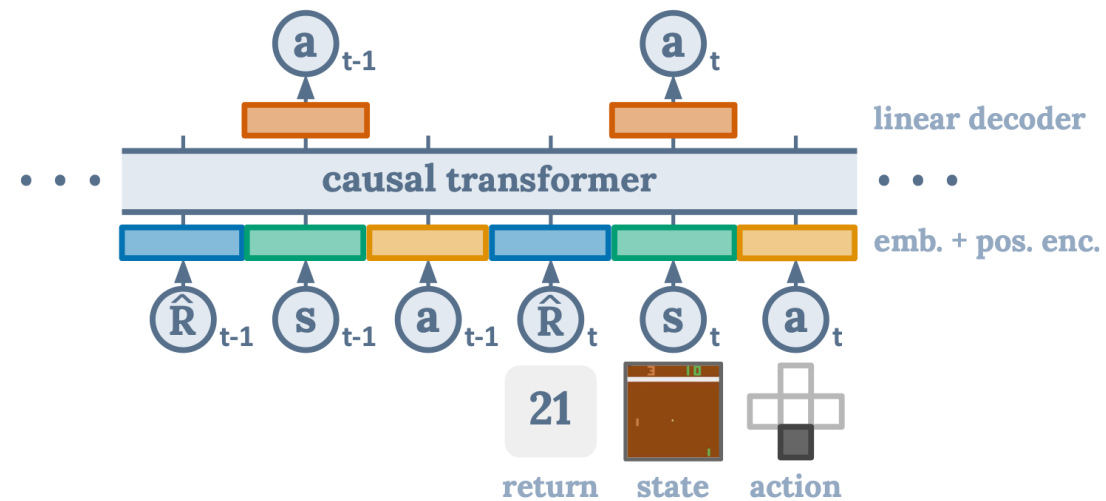
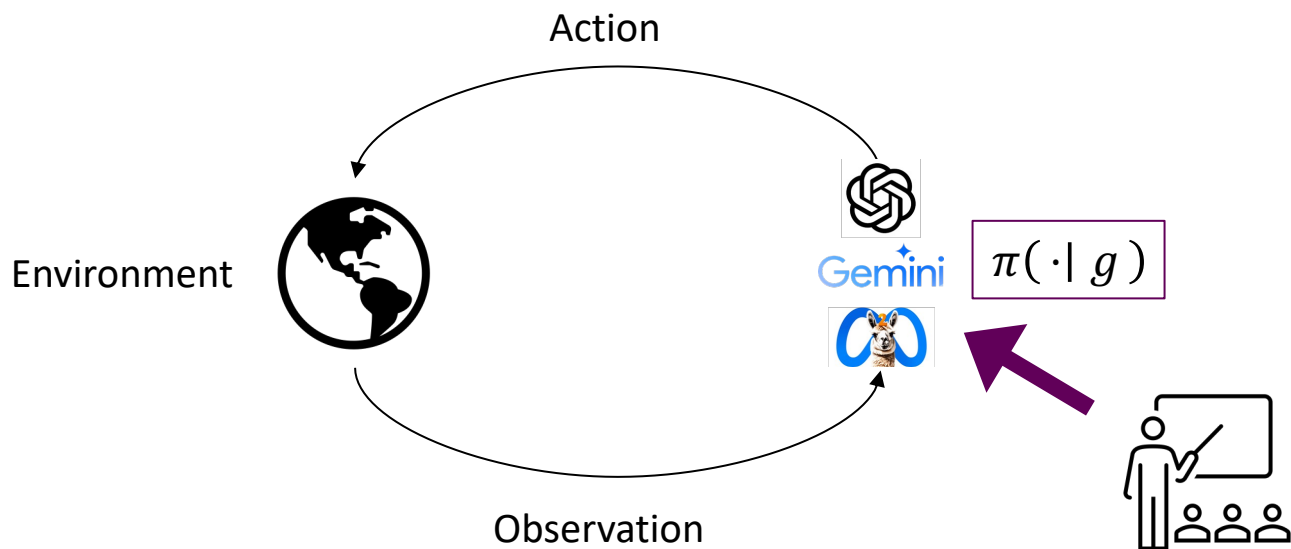


Idea #3: Use RL to directly map instructions to actions

$$\max_{\theta} \mathbb{E}_{a \sim \pi_{\theta}} R(a; \text{instruction}, \text{observation})$$



# Instruction following agents [in 2024]



Source: Chen et al. 2021

Main Idea: Use causal transformers for **generative trajectory modeling!**

$$p(\tau | g) = p(s_1, a_1, s_2, a_2, \dots | g) = \prod_t p(s_t | s_{t-1}, a_t) \times \pi(a_t | \tau_{\leq t}, g)$$

# A Simple Language Model Agent with ReACT

You are an agent capable of the following actions:

1. Type X on Y
2. Move mouse to
3. Click on X
4. Type Char x on Y

Your objective is to follow user instructions, by mapping them into a sequence of actions.

Instruction: {g}

So far, you have taken the following actions and observed the following environment states:

Previous Actions and Observations:

o1:

a1:

o2:

a2:

...

After executing these actions, you observe the following HTML state: <HTML state>

Now, think about your next action:

Thought: [model-pred]

Now, take an action:

Action: [model-pred]

1. Action space in text
2. Instruction in text
3. Previous observations and actions
4. Provide current observation [as text]

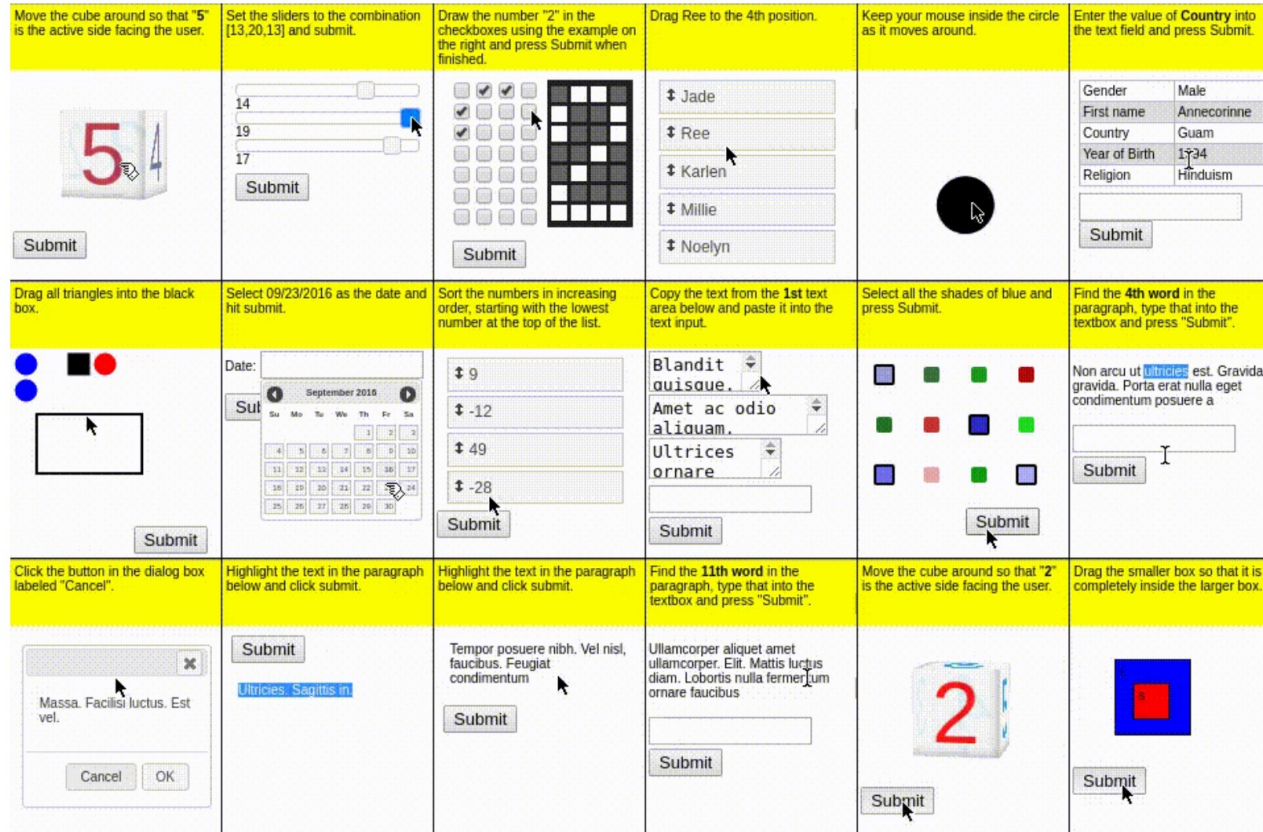
Model generates next action (sequence prediction task), use that action to update environment and repeat!

Mostly, just CoT prompting in a loop

$$\pi_{\text{LM}}(\cdot \mid g)$$

# Some popular benchmarks for LM agents:

## MiniWoB++



Sandboxed environment evaluating basic browser interactions across a range of applications from social media to email clients

Evaluates functional correctness


Not real world (limited functionality)

Relatively short-horizon


Zero-shot performance far from perfect!

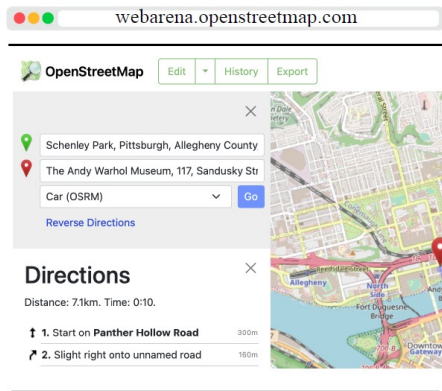
# Some popular benchmarks for LM agents:


## WebArena

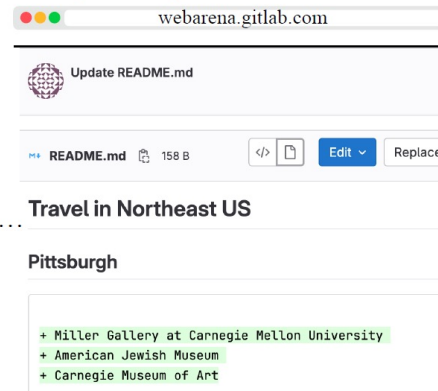
 “Create a plan to visit Pittsburgh's art museums with minimal driving distance starting from Schenley Park. Log the order in my “awesome-northeast-us-travel” repository ”




 Search for museums in Pittsburgh



 Search for each art museum on the Map



 Record the optimized results to the repo

Environment with sandboxed approximations of real websites spanning e-commerce, social media!

Additional utility tools: Maps, calculators, scratchpads, Wikipedia...

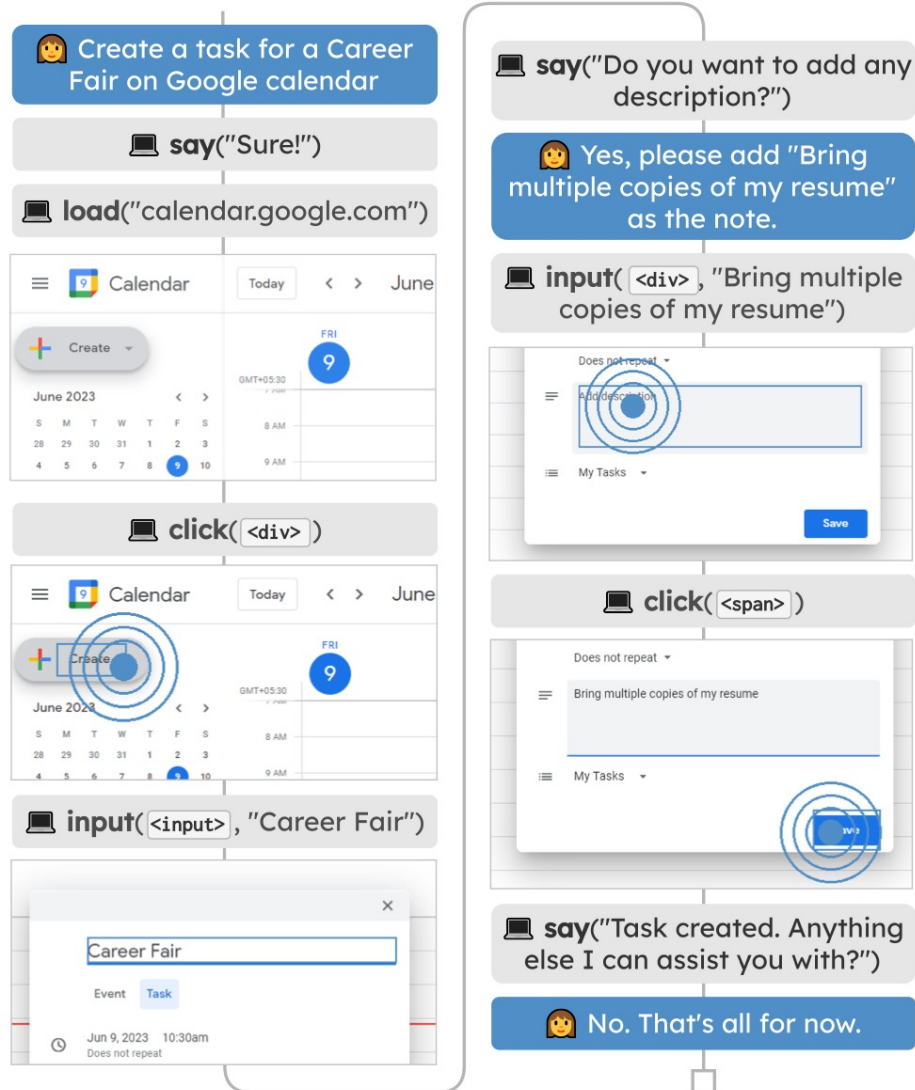
Multi-tab browsing

Long-horizon tasks

Evaluates functional correctness

# Some popular benchmarks for LM agents:

## WebLINX



Web-interactions on real websites

Conversational: includes a new “say” action to communicate with human to gather information

Multi-tab browsing

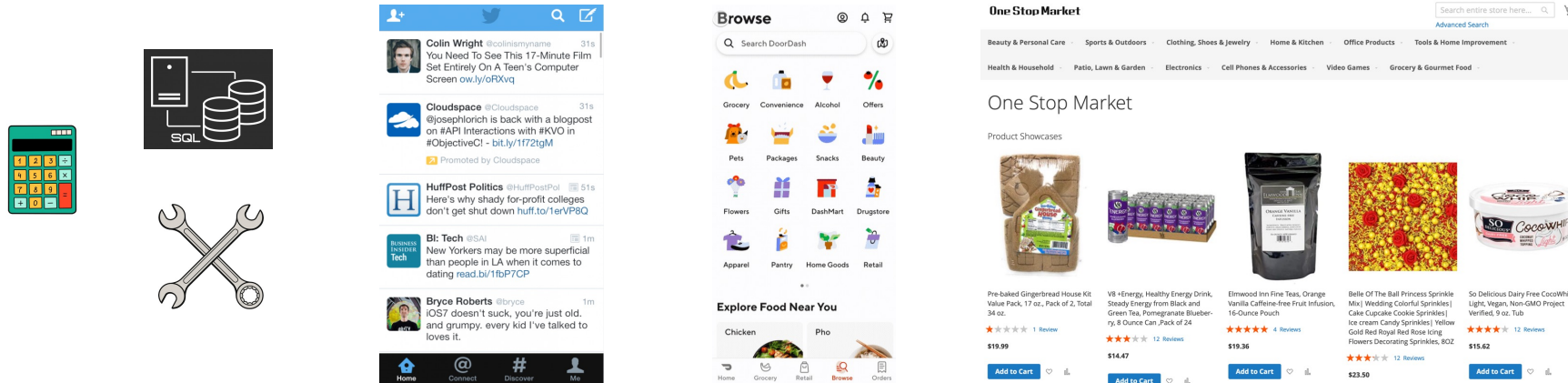
Turn-level metrics for evaluation

Not an environment, but a collection of interactions



# Training data for Language Model Agents

- Standard practice: In-context learning with few-shot demonstrations of humans performing following similar instructions.
- This is still not scalable / reliable



1000s of environments, many kinds of interactions possible...

Can agents autonomously explore their environments to construct high quality synthetic demonstrations?

# Use Exploration + Model Generated Data!

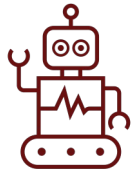
Book Your One-Way Flight

From:

To:

Departure Date

Search

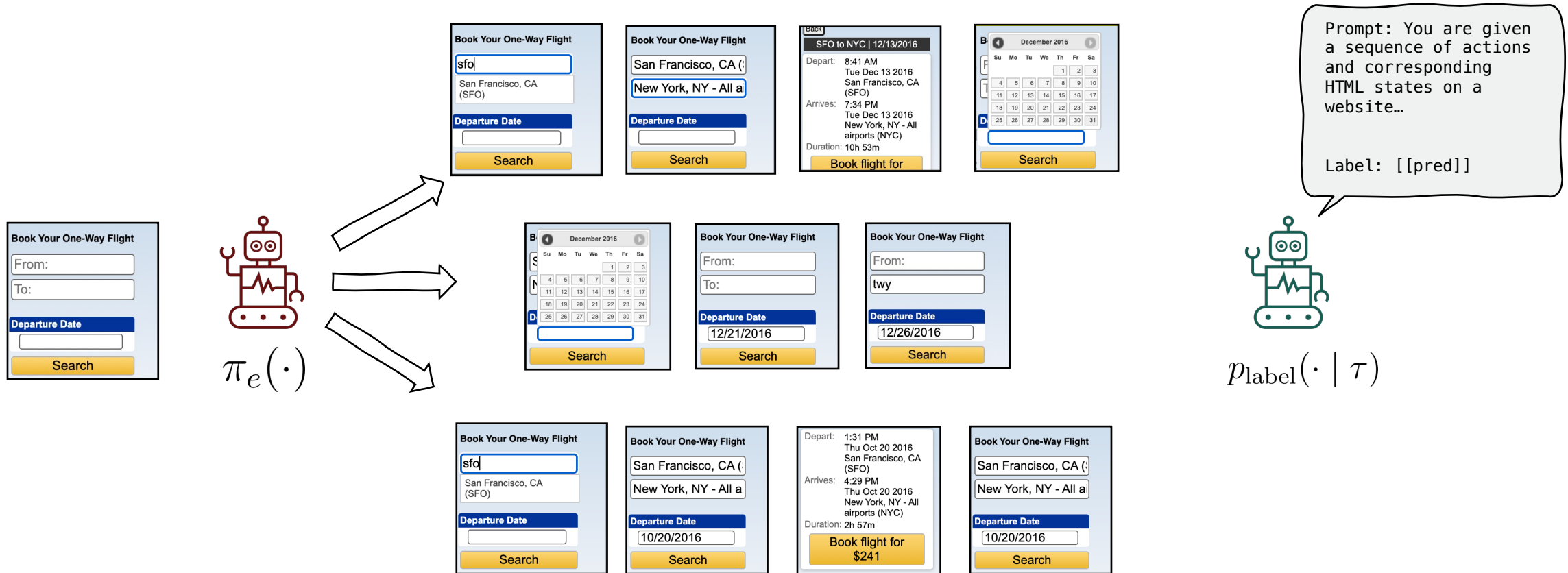


$\pi_e(\cdot)$

Prompt: Given a website, take actions of the following format to explore...

Action: [[pred]]

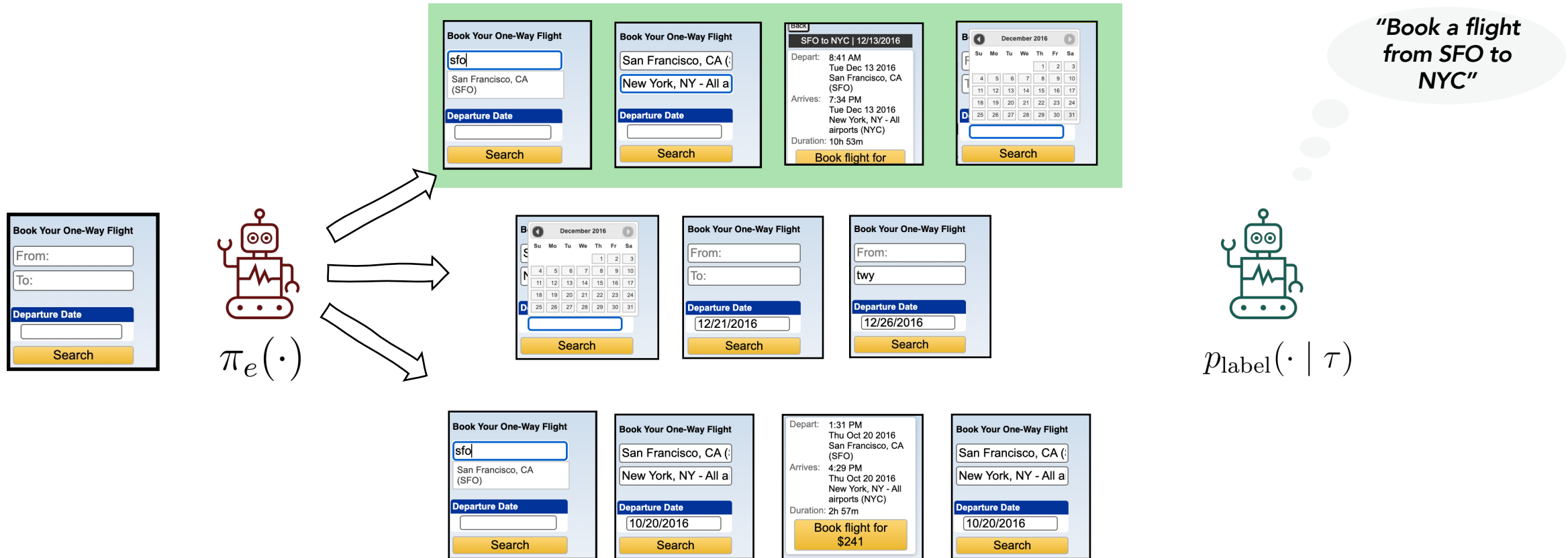
# Use Exploration + Model Generated Data!



How can we decide if a sequence of interactions is meaningful?  
*Use Natural Language!*

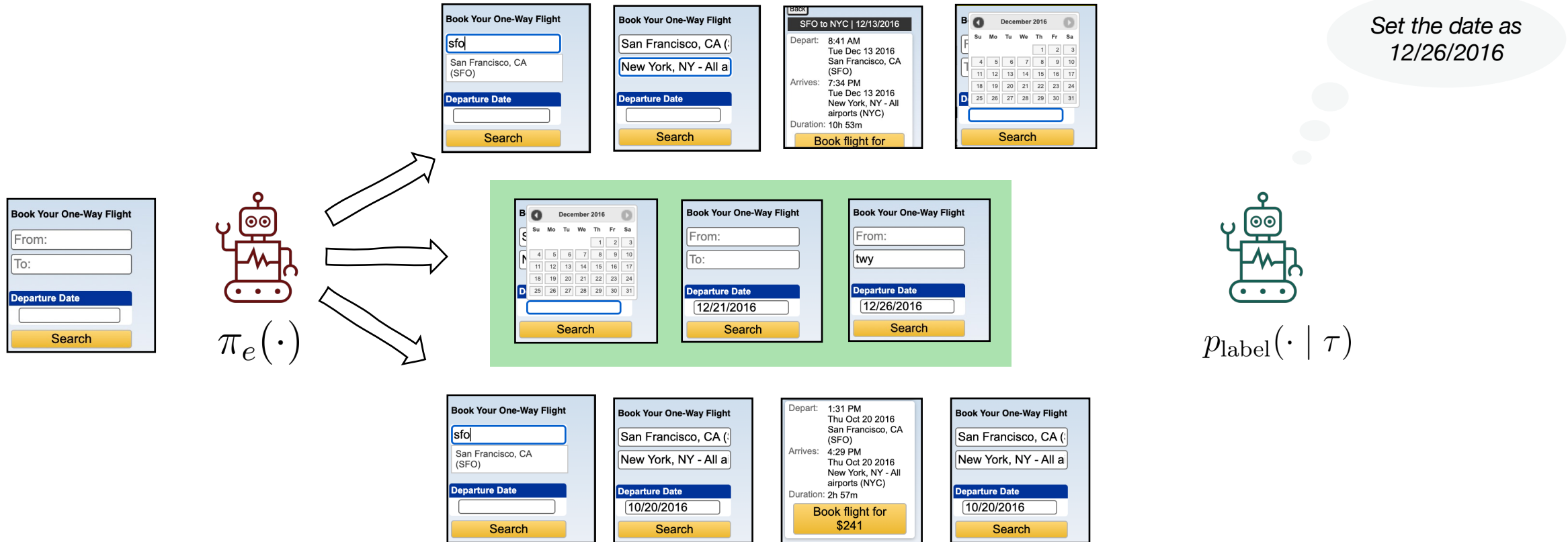


# Use Exploration + Model Generated Data!



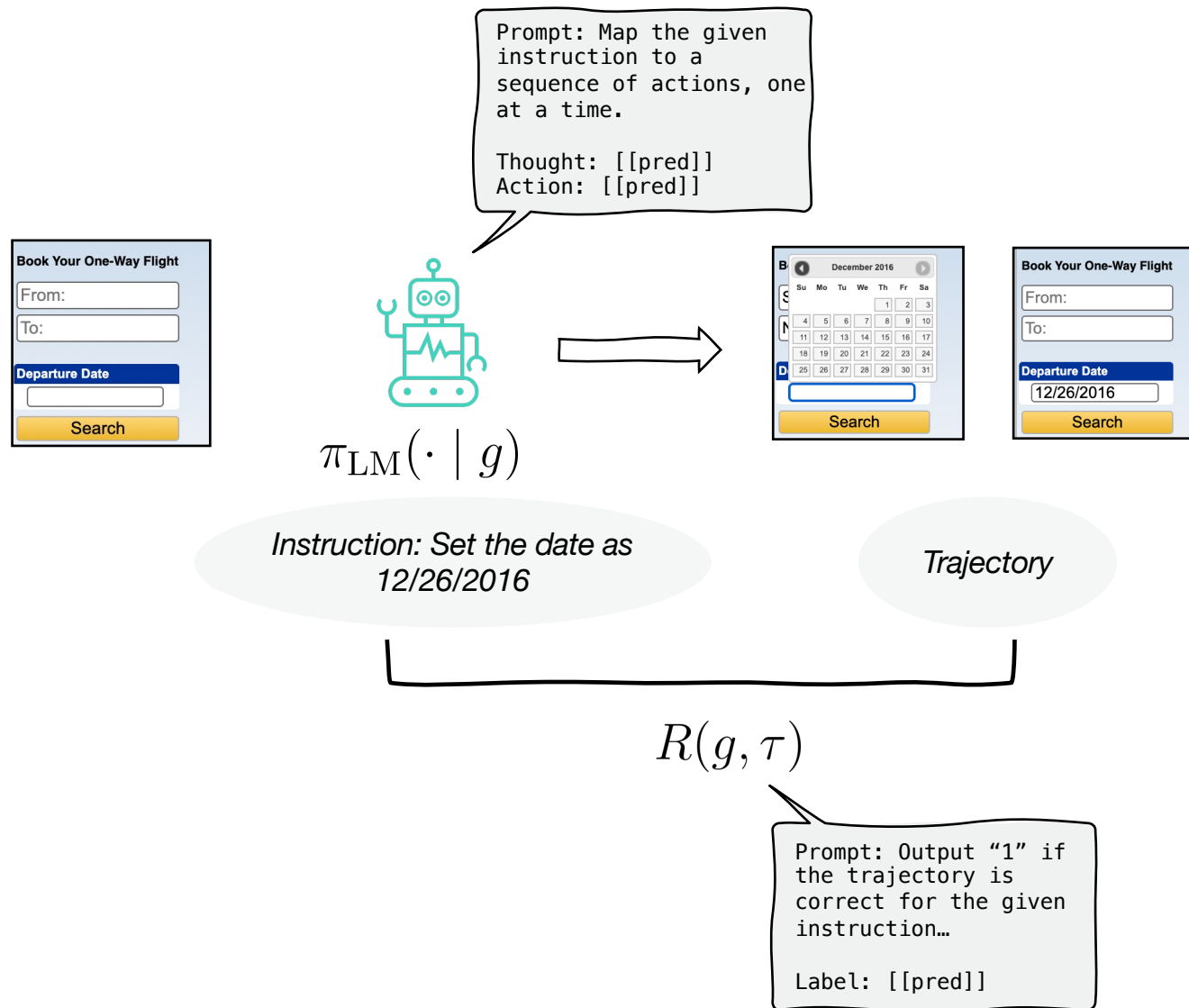
How can we decide if a sequence of interactions is meaningful?  
*Use Natural Language!*

# Use Exploration + Model Generated Data!

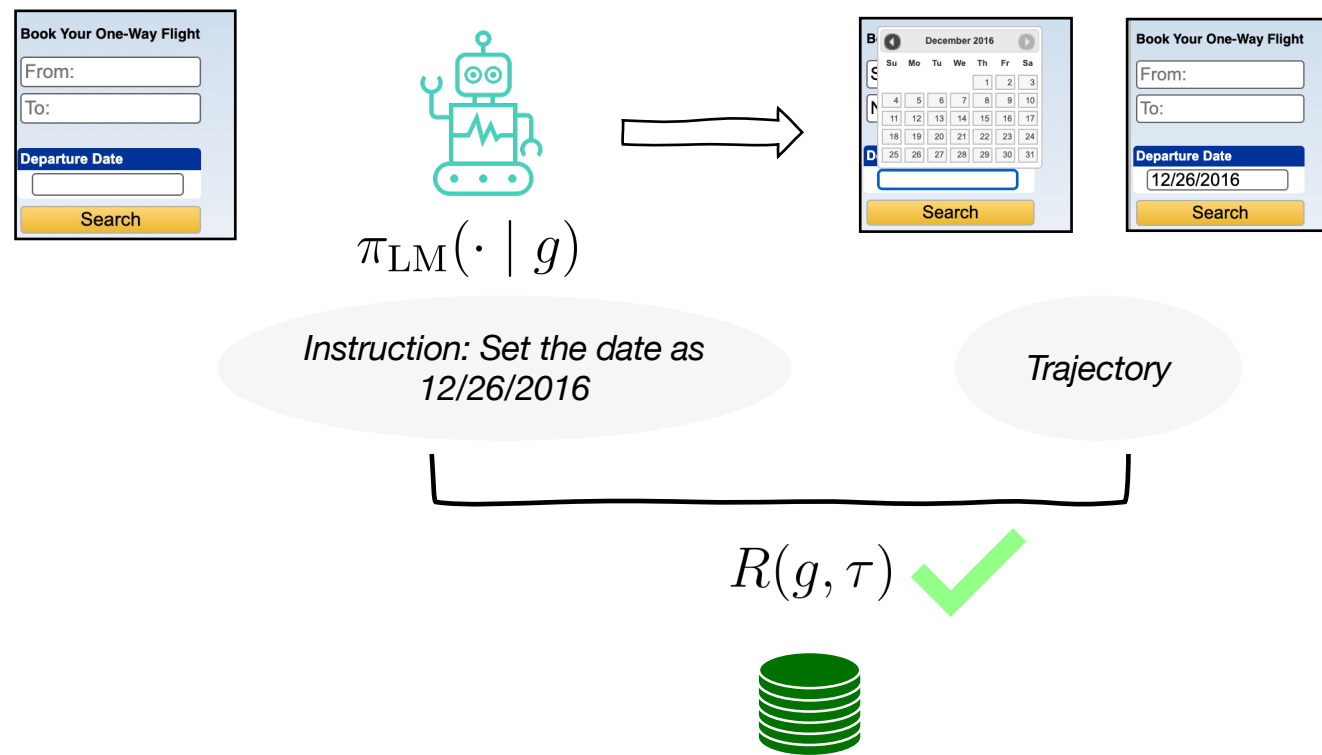


How can we decide if a sequence of interactions is meaningful?  
*Use Natural Language!*

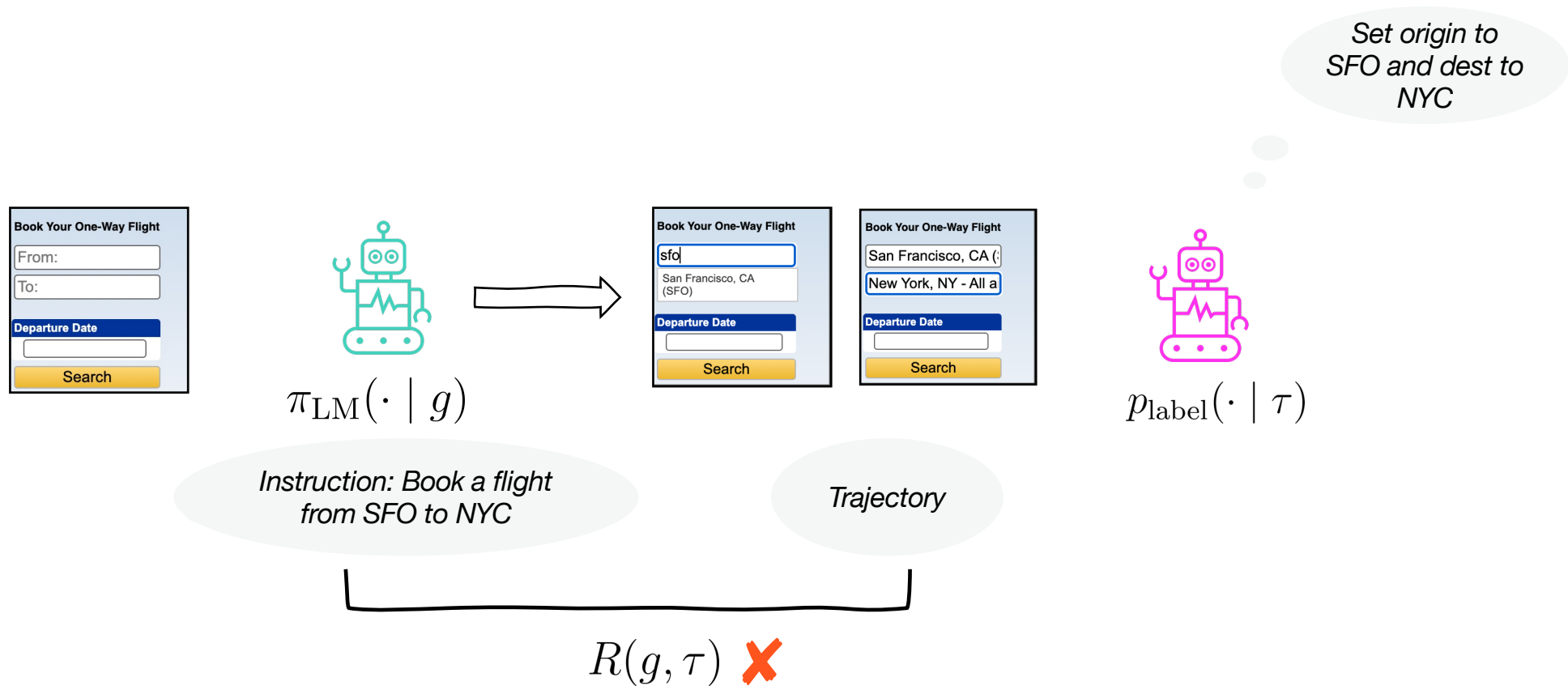
# Use Exploration + Model Generated Data!



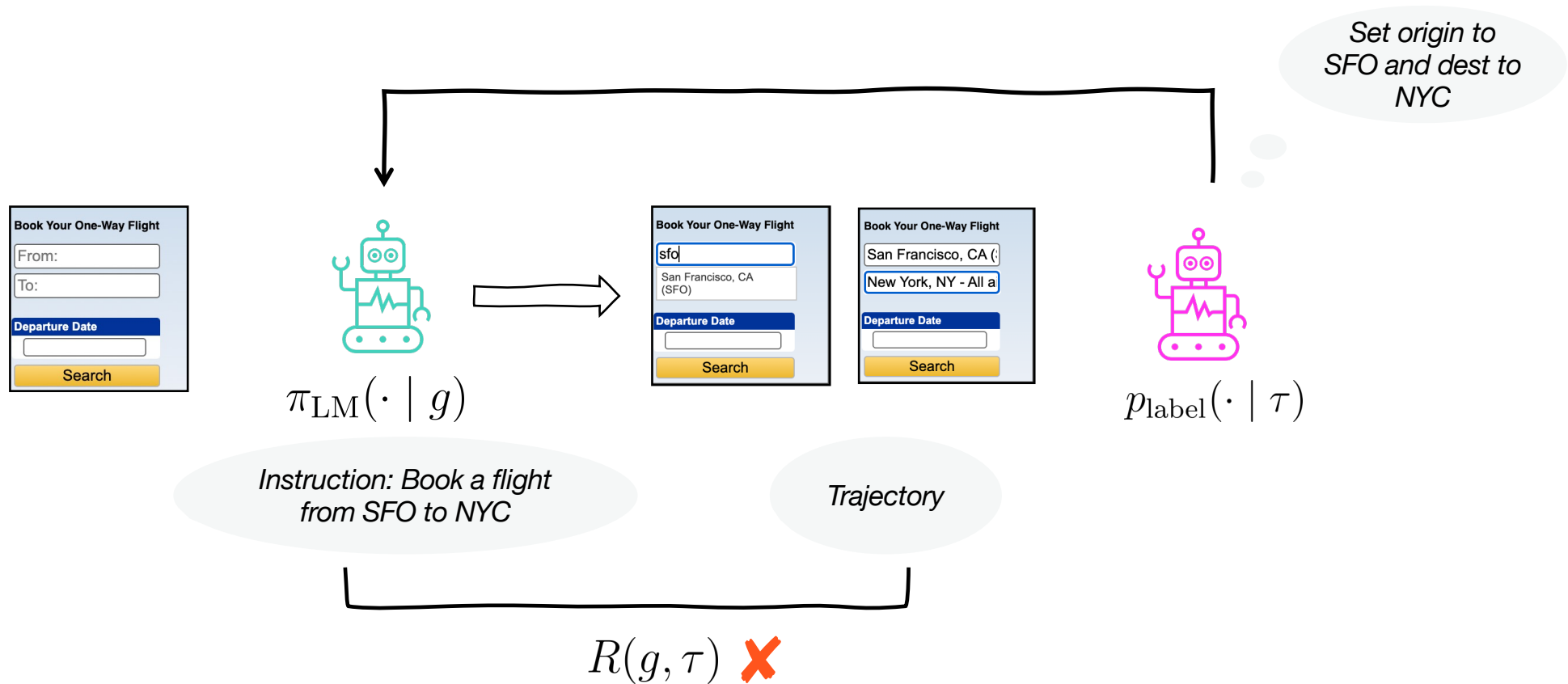
# Use Exploration + Model Generated Data!



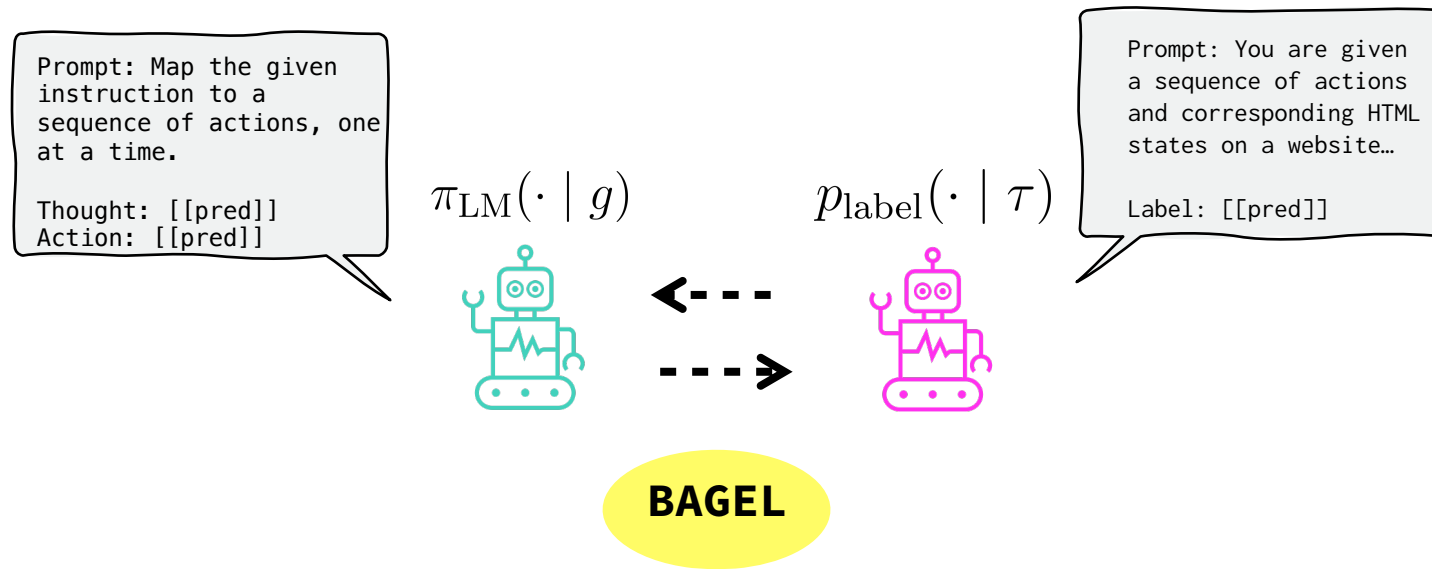
# Use Exploration + Model Generated Data!



# Use Exploration + Model Generated Data!

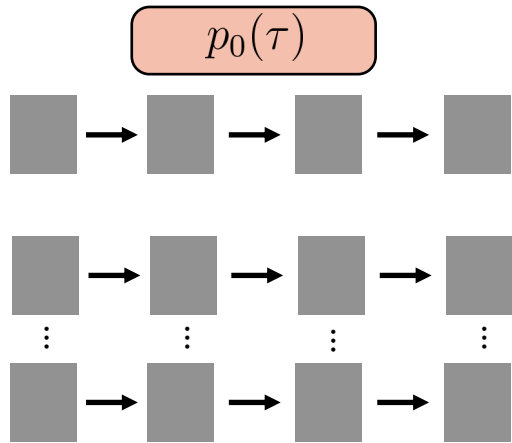


# BAGEL: Use Exploration + Model Generated Data!

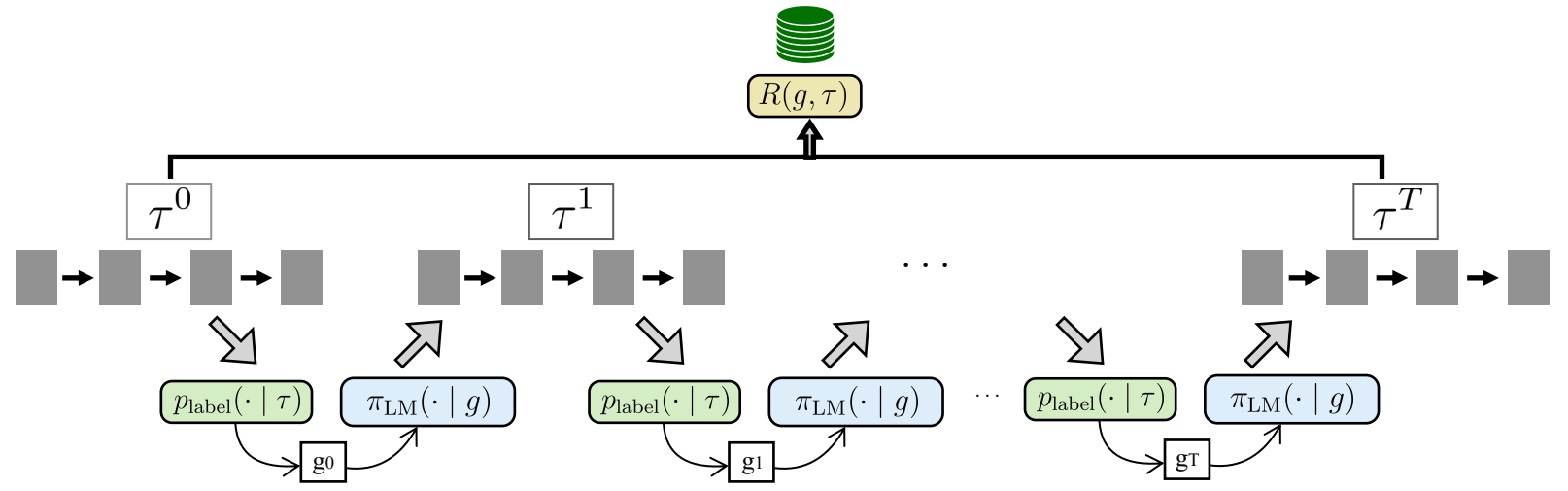


(Bootstrapping Agents by Guiding Exploration with Language)

# BAGEL: Use Exploration + Model Generated Data!



1. Explore Environment to collect trajectories



2. Create Synthetic demonstrations via iterative re-labeling

*Book the cheapest ...* → {type on ..., select ..., click ...}  
*Buy a flight from Denver ...* → {type ..., click ..., select ...}

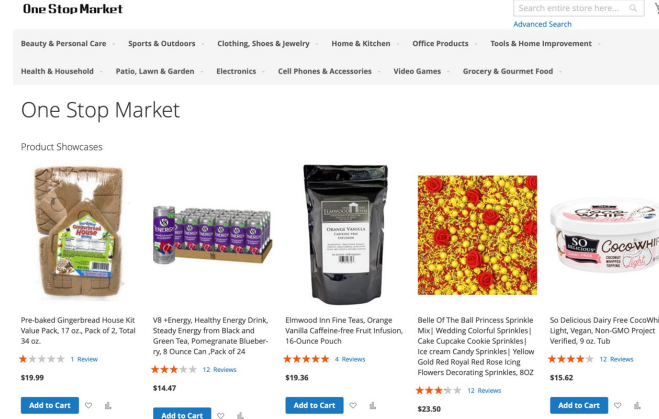
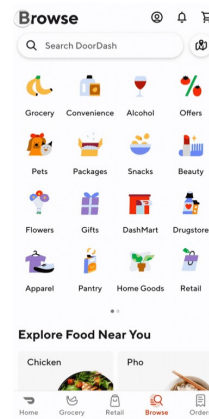
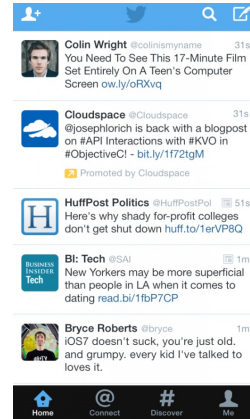
“Book the cheapest flight from Denver to LA”



3. Instruction-Following (Inference Time): Retrieve Relevant Demonstration via retrieval to use as in-context exemplars



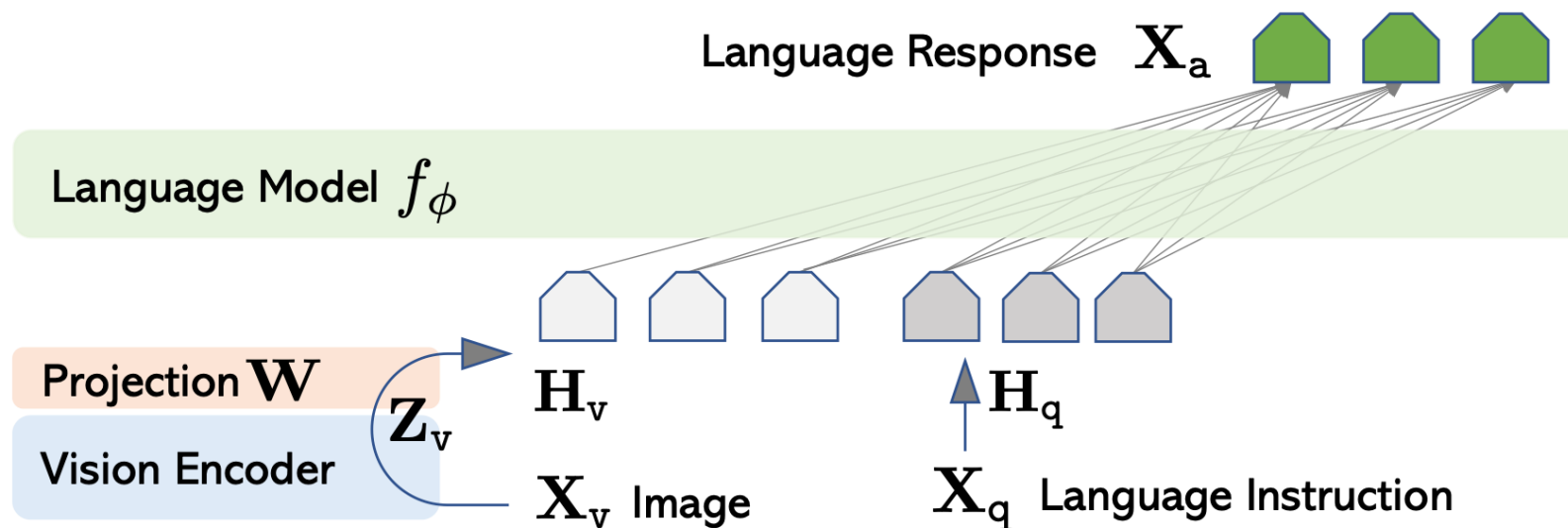
# Multimodality?



- So far, we've looked at using text-only language models for agents
- This is intractable for real-world UIs with very long HTML
- Can we instead operate directly over pixel space?

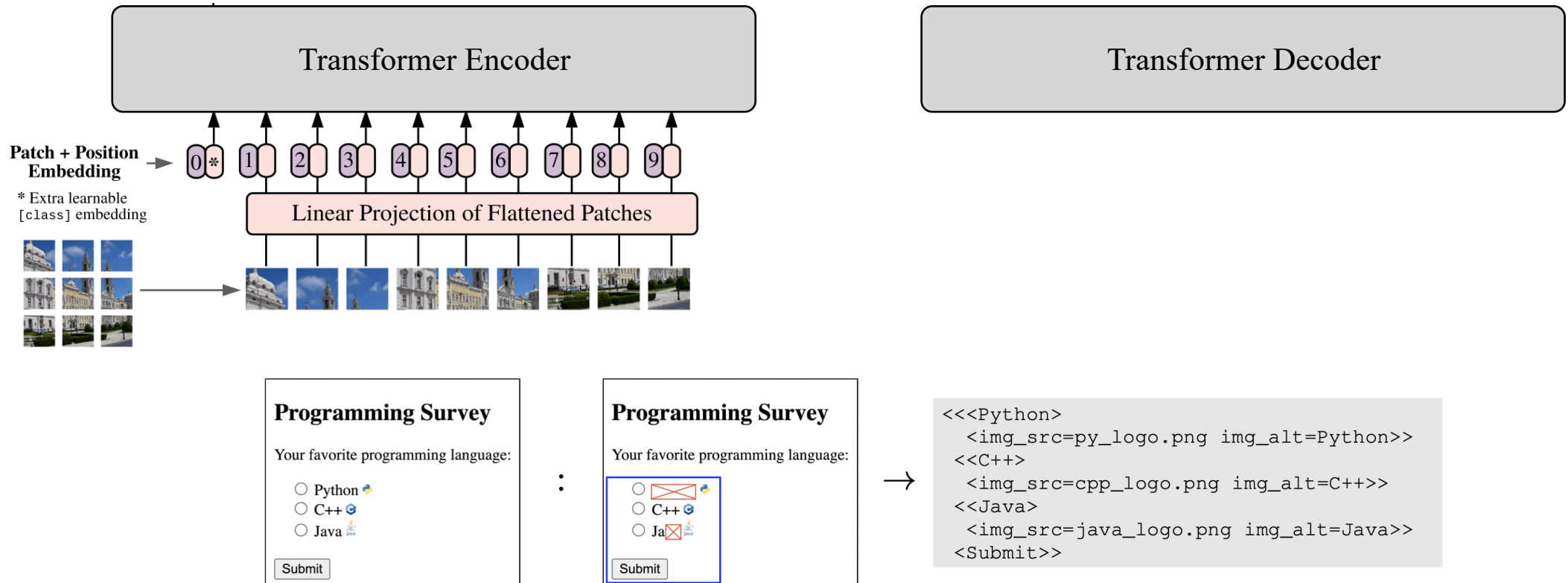
# Multimodality

## LLaVA

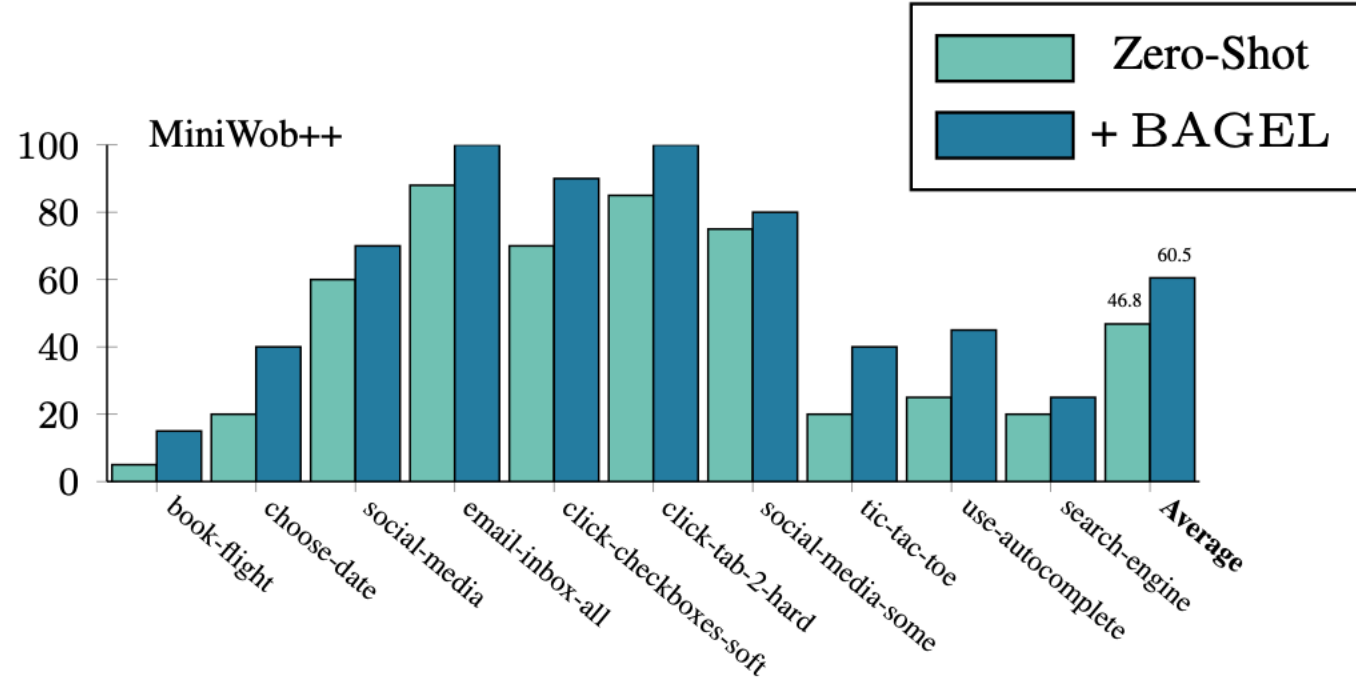


# Multimodality

## Pix2Struct

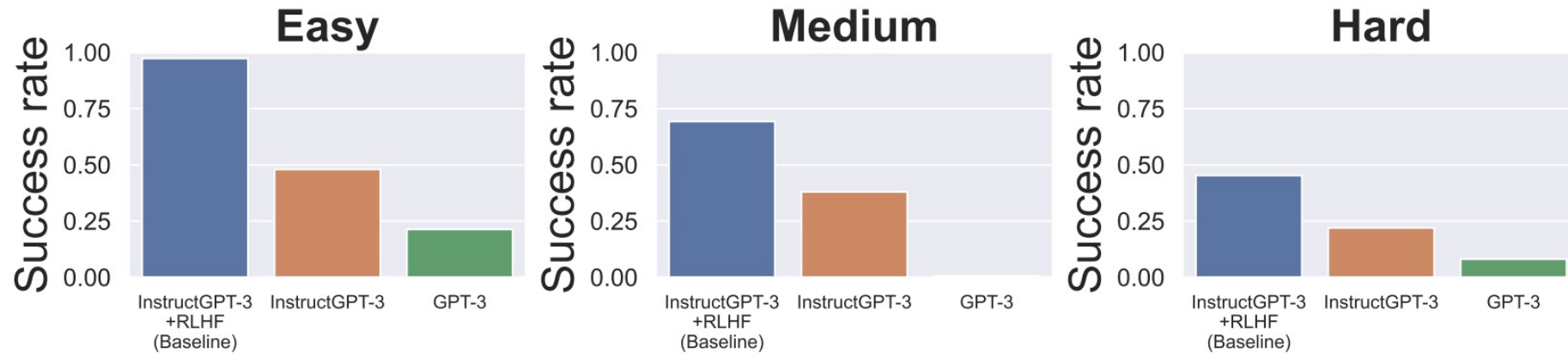


# LM Agents is an emerging application!



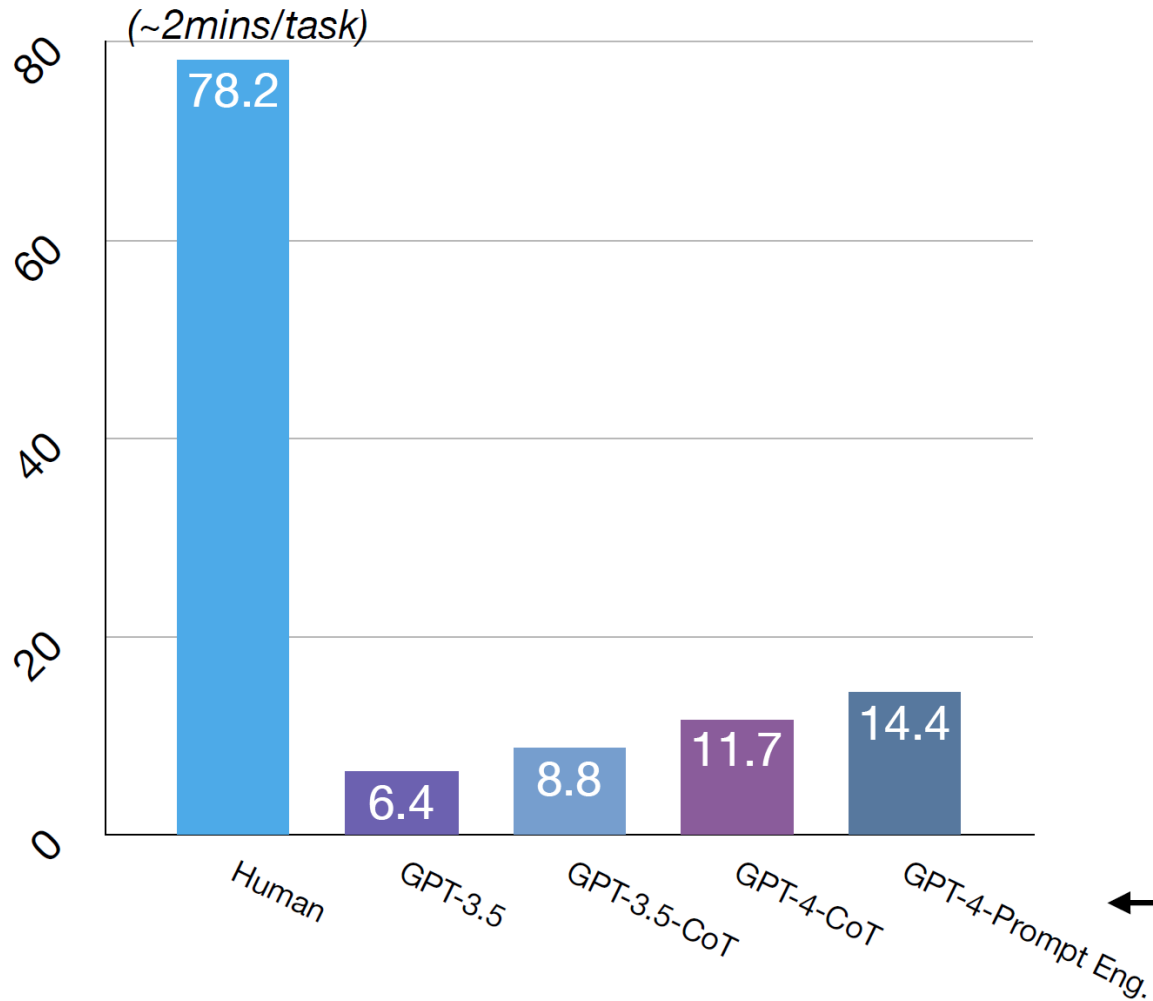
The “prompting gap”: without extensive prompting / bespoke few-shot examples, competitive LMs are far from perfect on even the simplest environments

# LM Agents is an emerging application!



Long-horizon planning is hard: Even on simple benchmarks, performance drops drastically on tasks that require longer horizon planning.

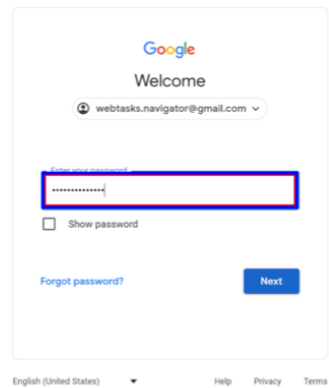
# LM Agents is an emerging application!



← Latest Work: BrowserGym 25%  
More prompt engineering  
More observation/action interface engineering

# LM Agents is an emerging application!

**S2:** Open Google translate and sign in using the following credentials: [email] [password]



**Reference (B):** [password]

**GPT-4V (R):** [email]

**LLaMA (B):** [password]



## Search

Search query

DMV area

Search

50 results for *DMV area*:

[2430] searchbox 'Search query'  
[5172] StaticText 'DMV area'

Search query

DMV areaDMV areaDMV areaDMV area

Search