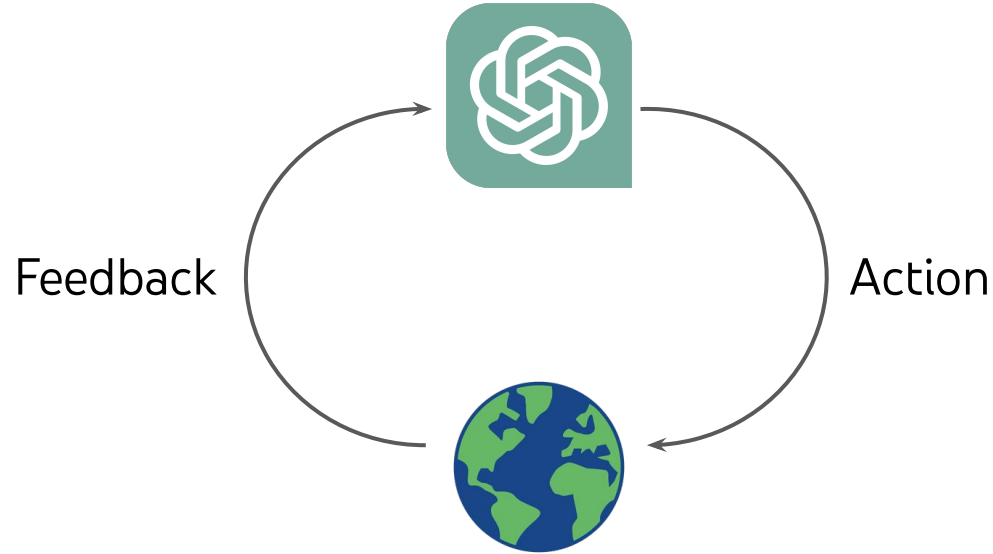




On Formulating and Evaluating Language Agents

Shunyu Yao

Language Agent



Use LLMs to interact with the world

A lot of Terms

- Language agent
- LLM-empowered agents
- LLM powered autonomous agents
- Language enabled agents
- LLM based agents
-

A lot of Papers

- SayCan
- ReAct
- Toolformer
- Generative Agents
- Tree of Thoughts
-

A lot of Products

- ChatGPT plugins
- Windows copilot
- Perplexity search
- LangChain
- Adept ACT-1
-

A lack of Theories

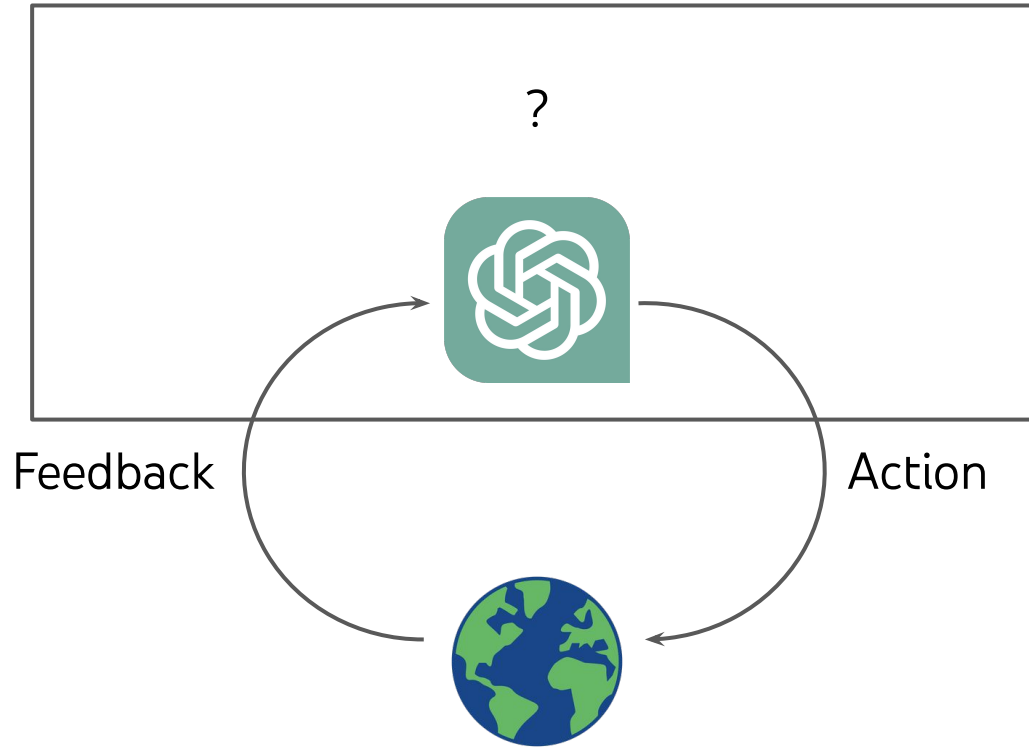
- What defines “language agent”?
- How to unify existing efforts?
- What is lacking?

A lack of Benchmarks

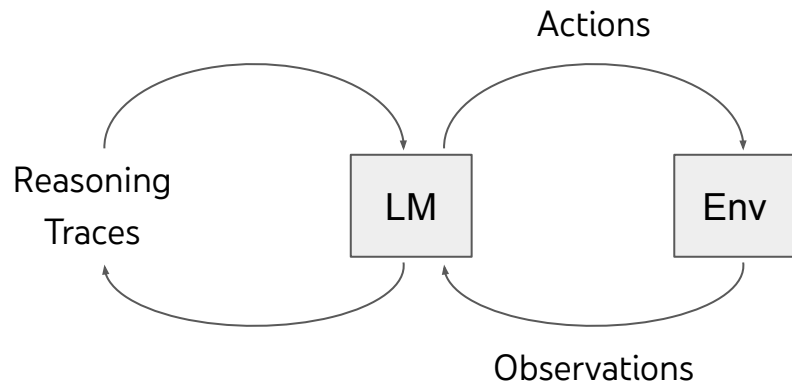
- NLP tasks? (too easy to solve?)
- Robotics tasks? (too hard to set up?)
- Evaluation? (too noisy and subjective?)

Part 1. Formulation

Language Agent



ReAct (Yao et al., 2022)



(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.

Act 1: `Search[Apple Remote]`

Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple ... originally designed to control the Front Row media center program ...

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.

Act 2: `Search[Front Row]`

Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports', 'Front Row (software)', ...]

Thought 3: Front Row is not found. I need to search Front Row (software) .

Act 3: `Search[Front Row (software)]`

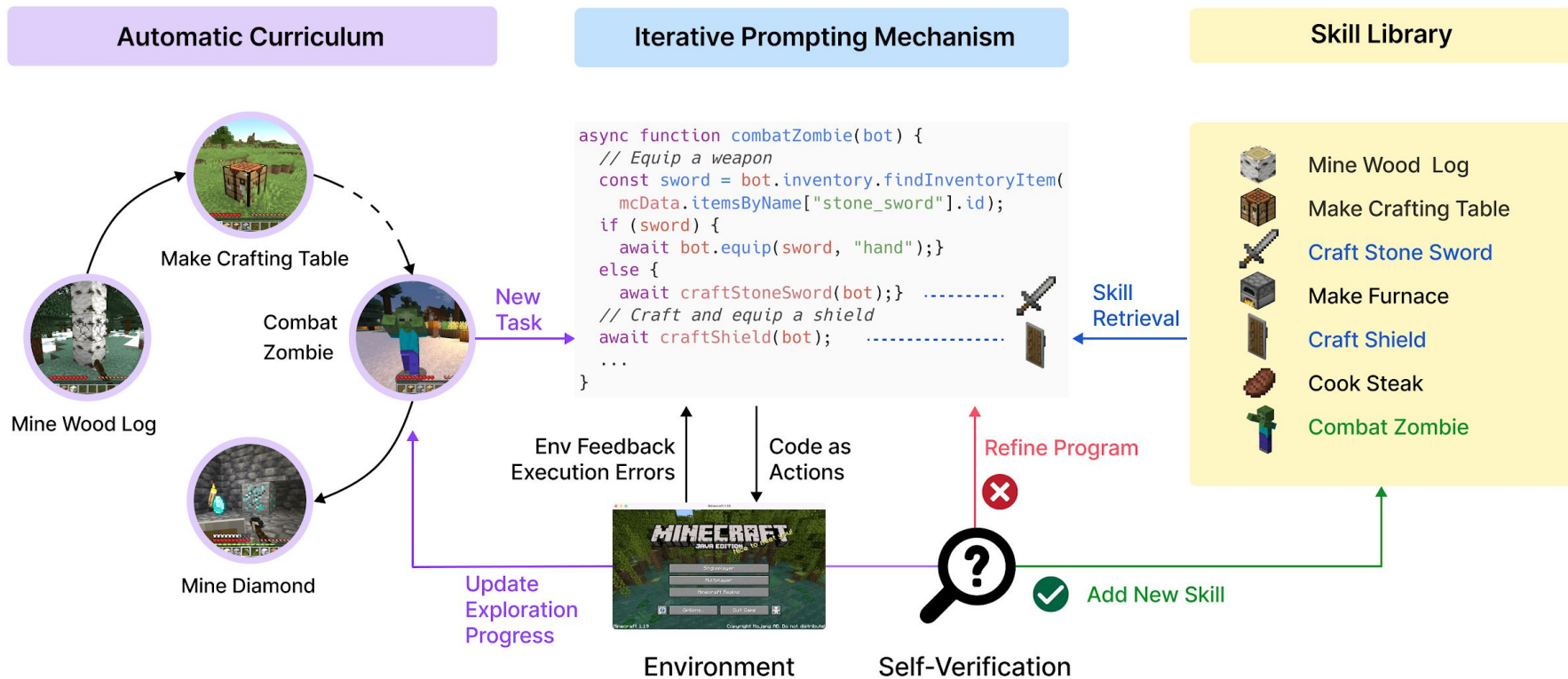
Obs 3: Front Row is a discontinued media center software ...

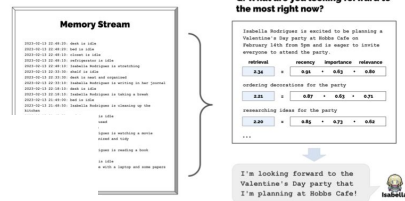
Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.

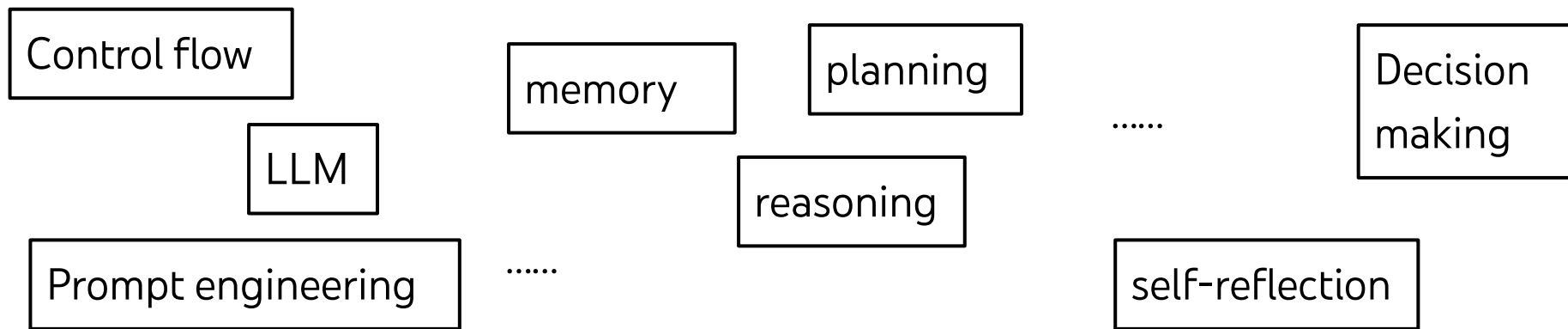
Act 4: `Finish[keyboard function keys]`



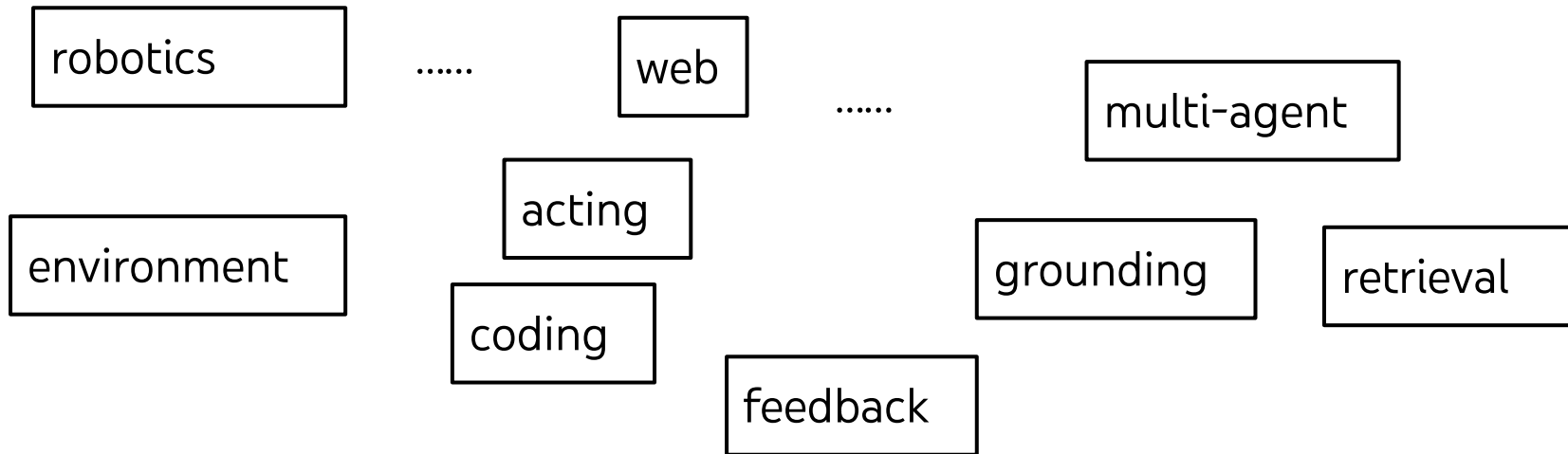
Voyager (Wang et al., 2023)







How to **make sense** of Language Agents?



Circuits

How to **make sense** of ~~Language Agents?~~

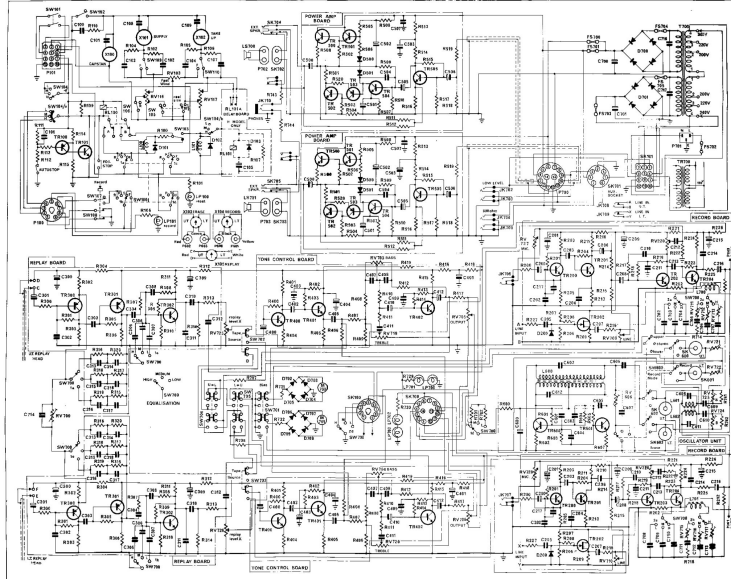
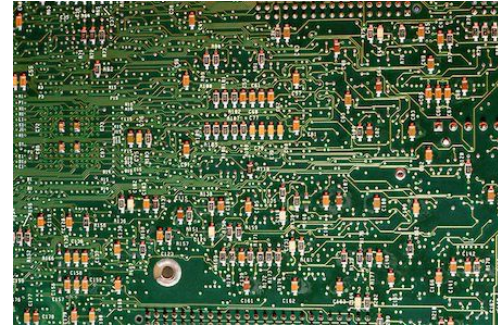
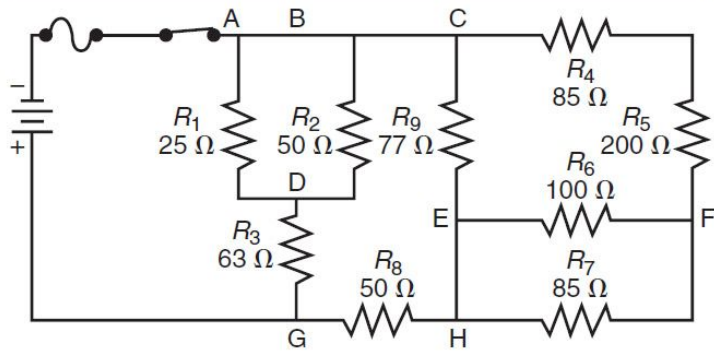
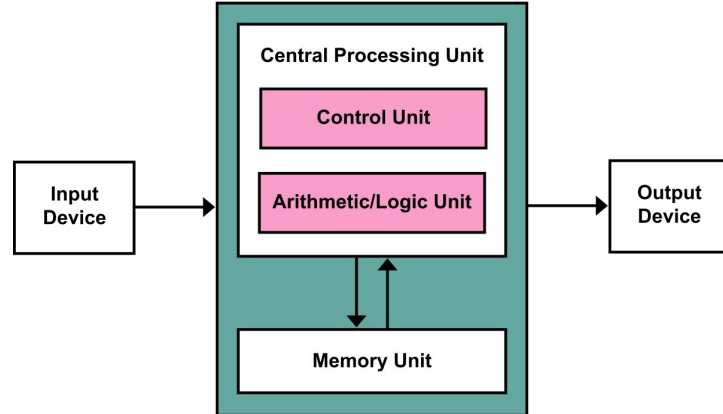


FIG 29. CIRCUIT DIAGRAM OF RECORDER

250-060 ISSUE 2



Von Neumann **architecture** makes sense & guides building of circuits.

How to **make sense** of Language Agents?

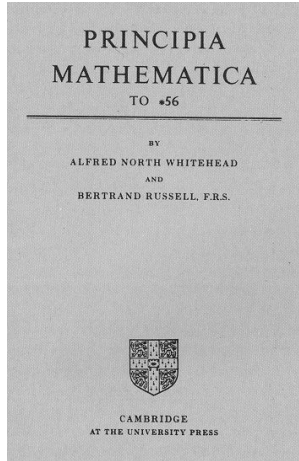
A system architecture, with LLM as a component
(computer architecture, with CPU as a component)

Cognitive Architectures for Language Agents

Theodore Sumers* Shunyu Yao* Karthik Narasimhan Thomas L. Griffiths
Princeton University
{sumers, shunyuy, karthikn, tomg}@princeton.edu

- **History:** What's cognitive architecture?
- **Analog:** Why it's related to language agents?
- **Framework:** How to formulate language agents via CoALA?
- **Insights:** Future directions through CoALA?

1900-1950: Production System Theorized



*54.43. $\vdash \therefore \alpha, \beta \in 1. \supset : \alpha \cap \beta = \Lambda. \equiv . \alpha \cup \beta \in 2$

Dem.

$\vdash . *54.26. \supset \vdash \therefore \alpha = \iota'x. \beta = \iota'y. \supset : \alpha \cup \beta \in 2. \equiv . x \neq y.$

$[*51.231] \quad \equiv . \iota'x \cap \iota'y = \Lambda.$

$[*13.12] \quad \equiv . \alpha \cap \beta = \Lambda \quad (1)$

$\vdash . (1). *11.11.35. \supset$

$\vdash \therefore (\exists x, y). \alpha = \iota'x. \beta = \iota'y. \supset : \alpha \cup \beta \in 2. \equiv . \alpha \cap \beta = \Lambda \quad (2)$

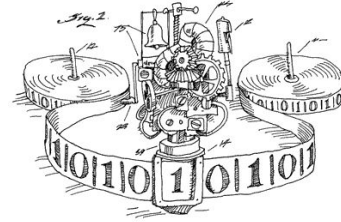
$\vdash . (2). *11.54. *52.1. \supset \vdash . \text{Prop}$

From this proposition it will follow, when arithmetical addition has been defined, that $1 + 1 = 2$.

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO
THE ENTSCHIEDUNGSPROBLEM

By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]



$0 = \lambda s. \lambda z. z$

$1 = \lambda s. \lambda z. sz$

$2 = \lambda s. \lambda z. s (s z)$

$3 = \lambda s. \lambda z. s (s (s z))$

A SET OF POSTULATES FOR THE FOUNDATION
OF LOGIC.¹

By ALONZO CHURCH.²

FORMAL REDUCTIONS OF THE GENERAL COMBINATORIAL
DECISION PROBLEM.*

By EMIL L. POST.

- **Symbol manipulation** formalizes math, logic, and computation
- **Production system** formalizes symbol manipulation:
 - a set of precondition \rightarrow action rules $X Y Z \rightarrow X W Z$
 - Also used to explain language & cognition

1950-1980: Production System Implemented

- Symbol manipulation -> **Physical symbol system**

$(\text{temperature} > 70^\circ) \wedge (\text{temperature} < 72^\circ) \rightarrow \text{stop}$

$\text{temperature} < 32^\circ \rightarrow \text{call for repairs; turn on electric heater}$

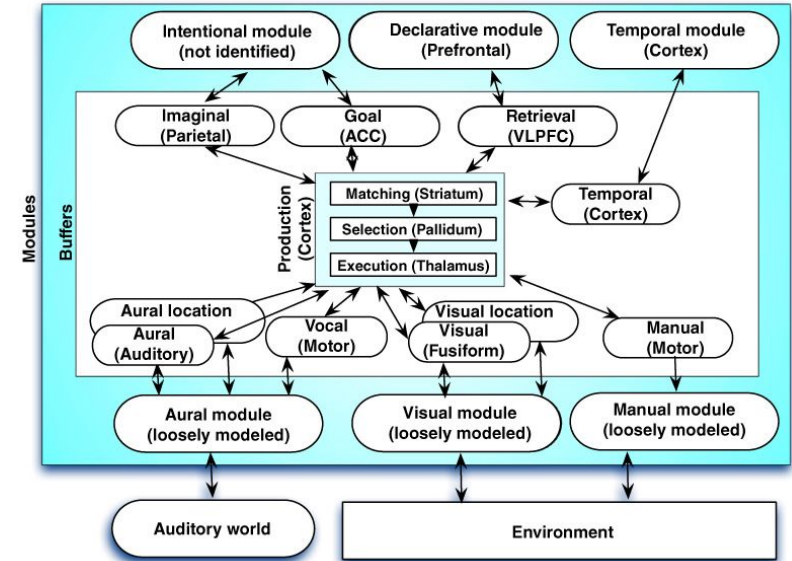
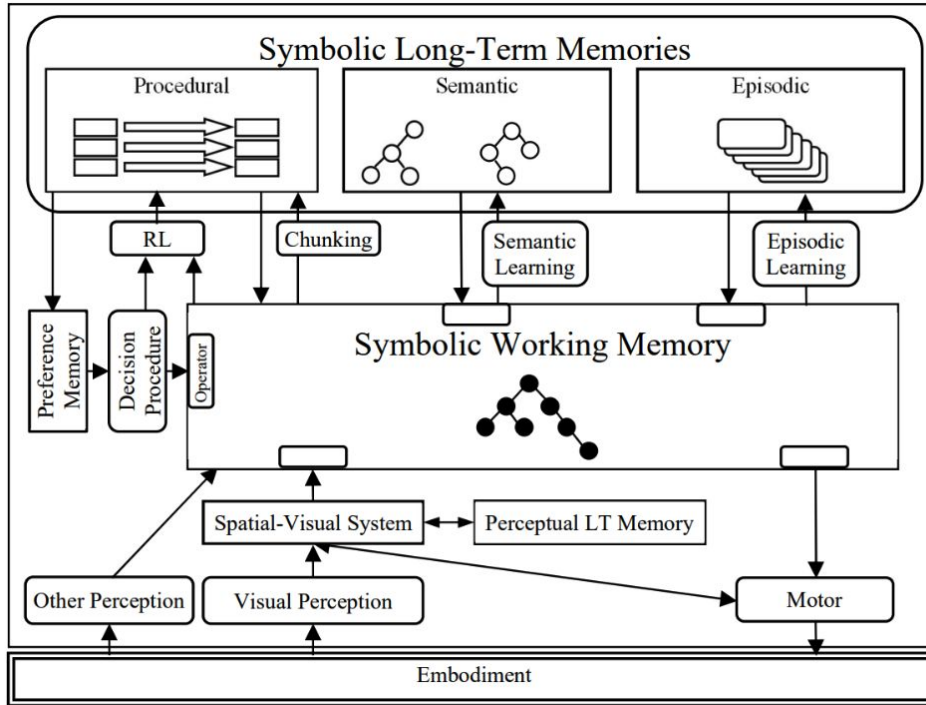
$(\text{temperature} < 70^\circ) \wedge (\text{furnace off}) \rightarrow \text{turn on furnace}$

$(\text{temperature} > 72^\circ) \wedge (\text{furnace on}) \rightarrow \text{turn off furnace}$

- **Usage towards real-world applications**

- Interact with the world → IO devices
- Many possible actions → Priorities over fired rules
- Complicated information → Memory mechanisms
- ...

Production System -> Cognitive Architectures



Issue: world is...

- **Complex:** too many rules
- **Stochastic:** rules can be fragile

2015 - 2022

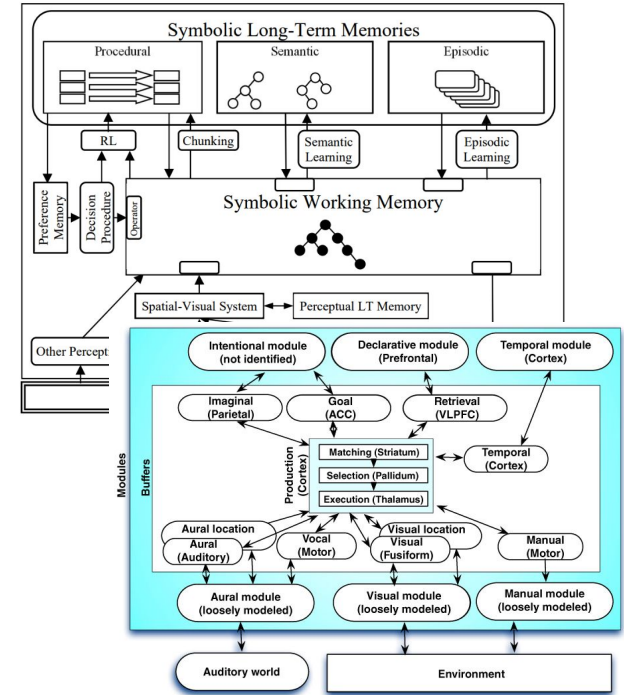
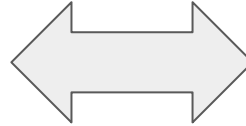
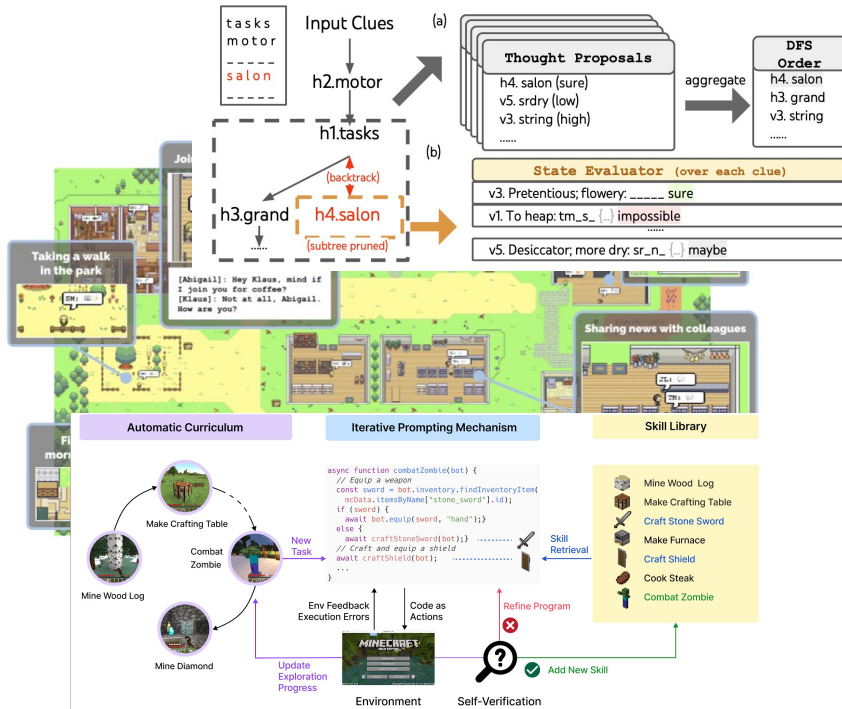


~~**Issue:** world is...~~

- ~~● **Complex:** too many rules~~
- ~~● **Stochastic:** rules can be fragile~~

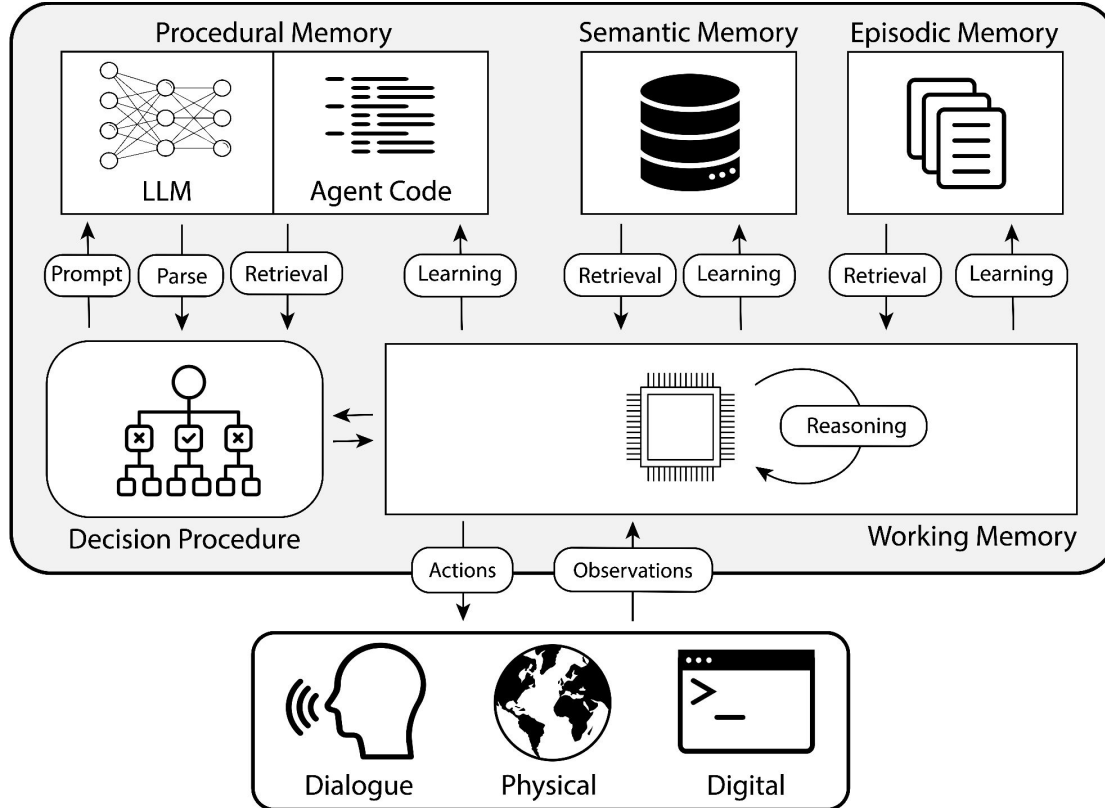
LMs are like large, implicit production systems

2022 - 2023



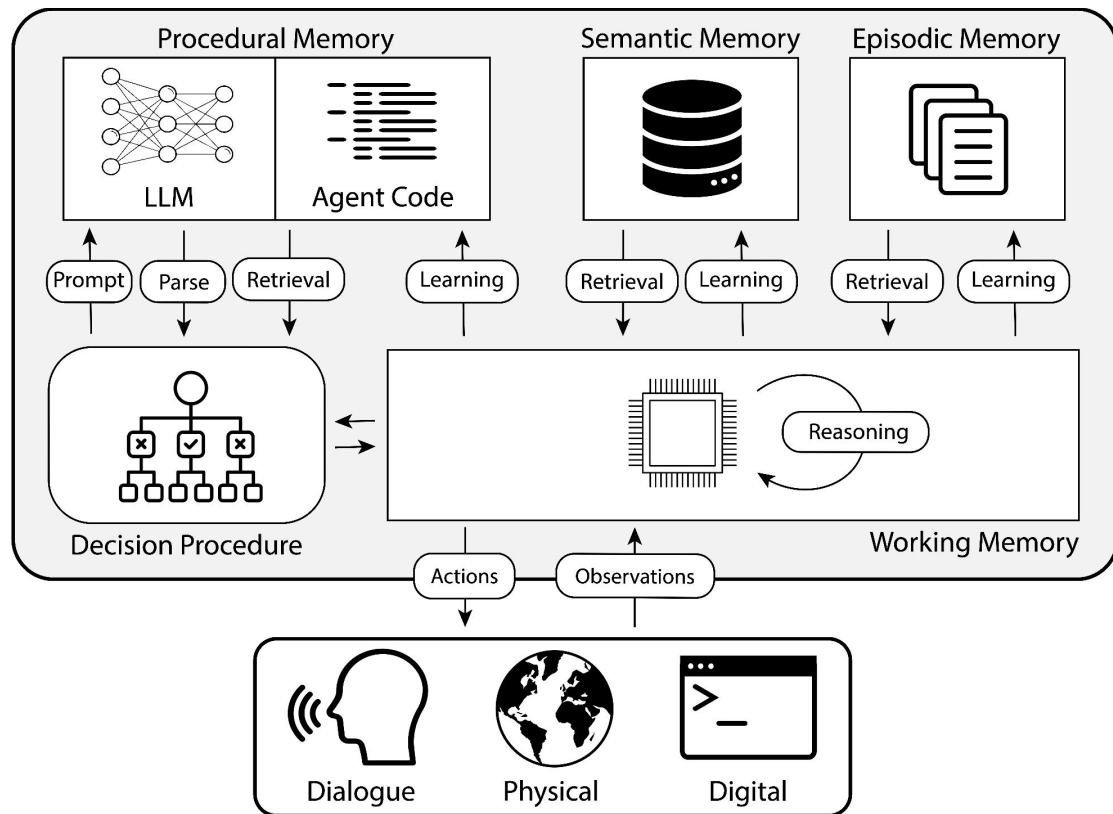
Language agents can be guided by Cognitive Architectures!

The CoALA framework



1. Memory
2. Action
3. Decision

1. Memory



Long-term memory

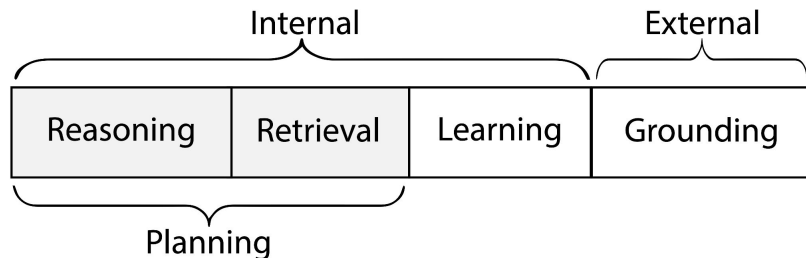
1. **Episodic** (experience)
2. **Semantic** (knowledge)
3. **Procedural** (LLM, code)

Short-term working memory

- Information for the current “decision cycle”

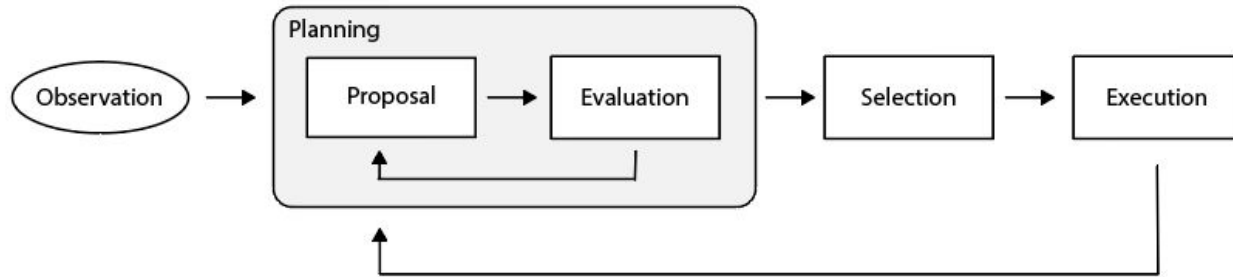
2. Action Space

- A language agent is defined with an **action space**
 - External actions interact with external environments (**grounding**)
 - Internal actions interact with internal memories
 - **Reasoning**: read & write working memory
 - **Retrieval**: read long-term memory
 - **Learning**: write long-term memory



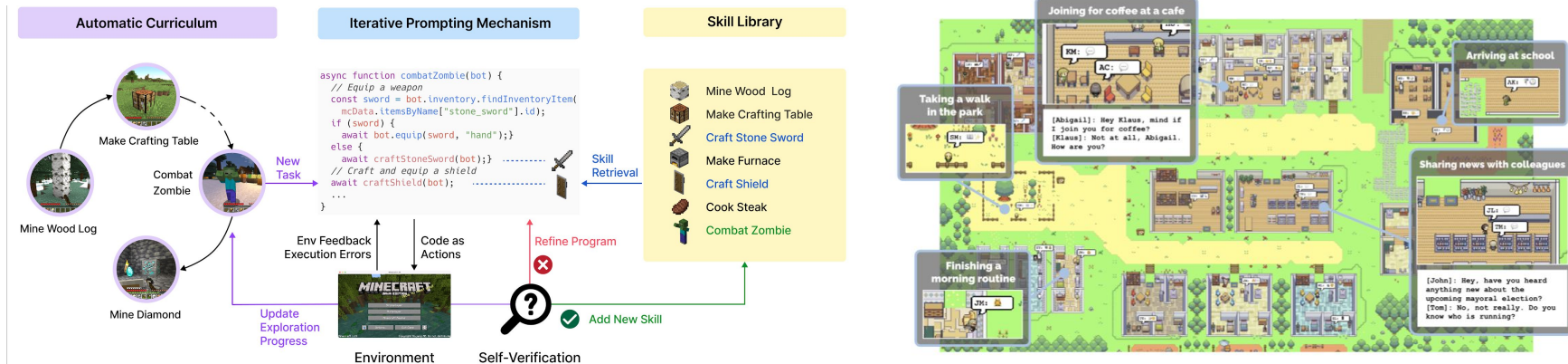
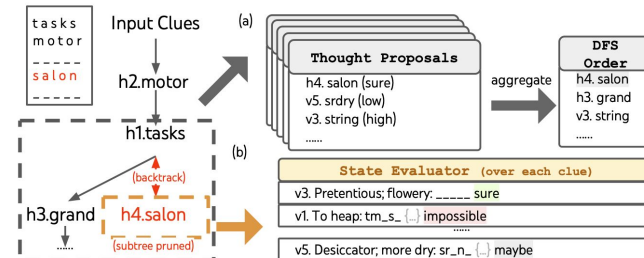
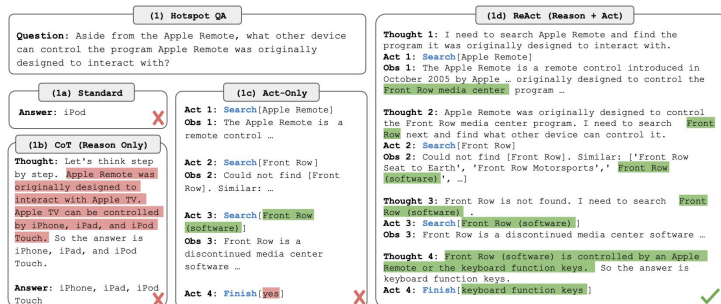
3. Decision Making

- A language agent chooses actions via **decision (making)** procedures
 - Split taken actions into decision cycles
 - In each cycle, plan then execute a learning/grounding action
 - **Planning:** use reasoning/retrieval to propose/evaluate actions
 - **Execution:** apply the learning/grounding action



That's it, basically.

Make Sense of (Existing) Language Agents



Make Sense of (Existing) Language Agents

	Long-term Memory ⁵	External Grounding	Internal Actions	Decision Making
SayCan (Ahn et al., 2022)	-	physical	-	evaluate
ReAct (Yao et al., 2022b)	-	digital	reason	propose
Voyager (Wang et al., 2023a)	procedural	digital	reason/retrieve/learn	propose
Generative Agents (Park et al., 2023)	episodic/semantic	digital/agent	reason/retrieve/learn	propose
Tree of Thoughts (Yao et al., 2023)	-	digital ⁶	reason	propose, evaluate, select

Make Sense of (Existing) Language Agents

Updating episodic memory with experience. It is common practice for RL agents to store episodic trajectories to update a parametric policy (Blundell et al., 2016; Pritzel et al., 2017) or establish a non-parametric policy (Ecoffet et al., 2019; Tuyls et al., 2022). For language agents, added experiences in episodic memory may be retrieved later as examples and bases for reasoning or decision making (Weston et al., 2014; Rubin et al., 2021; Park et al., 2023).

Updating semantic memory with knowledge. Recent work (Shinn et al., 2023; Park et al., 2023) has applied LLMs to reason about raw experiences and store the resulting inferences in semantic memory. For example, Reflexion (Shinn et al., 2023) uses an LLM to reflect on failed episodes and stores the results (e.g., “there is no dishwasher in kitchen”) as semantic knowledge to be attached to LLM context for solving later episodes. Finally, work in robotics (Chen et al., 2023a) uses vision-language models to build a semantic map of the environment, which can later be queried to execute instructions.

Updating LLM parameters (procedural memory). The LLM weights represent implicit procedural knowledge. These can be adjusted to an agent’s domain by fine-tuning during the agent’s lifetime. Such fine-tuning can be accomplished via supervised or imitation learning (Hussein et al., 2017), reinforcement learning (RL) from environment feedback (Sutton and Barto, 2018), human feedback (RLHF) (Christiano et al., 2017; Ouyang et al., 2022; Nakano et al., 2021), or AI feedback (Bai et al., 2022). For example, GTX (Tuyls et al., 2022) periodically finetunes a small language model on high-scoring trajectories stored in episodic memory, which serves as a robust “exploitation” policy to reach exploration frontiers in the face of stochasticity. Recent work (Huang et al., 2022a; Zelikman et al., 2022) has also shown the potential of finetuned small language models distilling then surpassing larger ones. Fine-tuning the agent’s LLM is a costly form of learning; thus, present studies specify learning schedules. However, as training becomes more efficient – or if agents utilize smaller subtask-specific LLMs – it may be possible to allow language agents to autonomously determine when and how to fine-tune their LLMs.

Updating agent code (procedural memory). CoALA allows agents to update their source code, thus modifying the implementation of various procedures. These can be broken down as follows:

- **Updating reasoning** (e.g., prompt templates; Gao et al., 2020; Zhou et al., 2022b). For example, APE (Zhou et al., 2022b) infers prompt instructions from input-output examples, then uses these instructions as part of the LLM prompt to assist task solving. Such a prompt update can be seen as a form of learning to reason.
- **Updating grounding** (e.g., code-based skills; Liang et al., 2023a; Ellis et al., 2021; Wang et al., 2023a). For example, Voyager (Wang et al., 2023a) maintains a curriculum library. Notably, current methods are limited to creating new code skills to interact with external environments.
- **Updating retrieval.** To our knowledge, these learning options are not studied in recent language agents. Retrieval is usually considered a basic action designed with some fixed implementation (e.g., BM25 or dense retrieval), but research in query/document expansion (Nogueira et al., 2019; Wang et al., 2023c; Tang et al., 2023a) or retrieval distillation (Izacard et al., 2021) may be helpful for language agents to learn better retrieval procedures.
- **Updating learning or decision-making.** Finally, it is theoretically possible for CoALA agents to learn new procedures for learning or decision making, thus providing significant adaptability. In general, however, updates to these procedures are risky both for the agent’s functionality and alignment. At present, we are not aware of any language agents that implement this form of learning; we discuss such possibilities more in Section 6.

While RL agents usually fix one way of learning (e.g., Q-learning, PPO, or A3C) and learn by updating model parameters, language agents can select from a diversity of learning procedures. This allows them to learn rapidly by storing task-relevant language (cheaper and quicker than parameter updates), and leverage multiple forms of learning to compound their self-improvement (e.g., Generative Agents discussed in Section 5).

Finally, while our discussion has mostly focused on adding to memory, modifying and deleting (a case of “unlearning”) are understudied in recent language agents. We address these areas more in Section 6.

Learn = Write long-term memory

- Unify various things

Guide Building of (Future) Language Agents

Updating episodic memory with experience. It is common practice for RL agents to store episodic trajectories to update a parametric policy (Blundell et al., 2016; Pritzel et al., 2017) or establish a non-parametric policy (Ecoffet et al., 2019; Tuyls et al., 2022). For language agents, added experiences in episodic memory may be retrieved later as examples and bases for reasoning or decision making (Weston et al., 2014; Rubin et al., 2021; Park et al., 2023).

Updating semantic memory with knowledge. Recent work (Shinn et al., 2023; Park et al., 2023) has applied LLMs to reason about raw experiences and store the resulting inferences in semantic memory. For example, Reflexion (Shinn et al., 2023) uses an LLM to reflect on failed episodes and stores the results (e.g., “there is no dishwasher in kitchen”) as semantic knowledge to be attached to LLM context for solving later episodes. Finally, work in robotics (Chen et al., 2023a) uses vision-language models to build a semantic map of the environment, which can later be queried to execute instructions.

Updating LLM parameters (procedural memory). The LLM weights represent implicit procedural knowledge. These can be adjusted to an agent’s domain by fine-tuning during the agent’s lifetime. Such fine-tuning can be accomplished via supervised or imitation learning (Hussein et al., 2017), reinforcement learning (RL) from environment feedback (Sutton and Barto, 2018), human feedback (RLHF) (Christiano et al., 2017; Ouyang et al., 2022; Nakano et al., 2021), or AI feedback (Bai et al., 2022). For example, GTX (Tuyls et al., 2022) periodically finetunes a small language model on high-scoring trajectories stored in episodic memory, which serves as a robust “exploitation” policy to reach exploration frontiers in the face of stochasticity. Recent work (Huang et al., 2022a; Zelikman et al., 2022) has also shown the potential of finetuned small language models distilling then surpassing larger ones. Fine-tuning the agent’s LLM is a costly form of learning; thus, present studies specify learning schedules. However, as training becomes more efficient – or if agents utilize smaller subtask-specific LLMs – it may be possible to allow language agents to autonomously determine when and how to fine-tune their LLMs.

Updating agent code (procedural memory). CoALA allows agents to update their source code, thus modifying the implementation of various procedures. These can be broken down as follows:

- **Updating reasoning** (e.g., prompt templates; Gao et al., 2020; Zhou et al., 2022b). For example, APE (Zhou et al., 2022b) infers prompt instructions from input-output examples, then uses these instructions as part of the LLM prompt to assist task solving. Such a prompt update can be seen as a form of learning to reason.
- **Updating grounding** (e.g., code-based skills; Liang et al., 2023a; Ellis et al., 2021; Wang et al., 2023a). For example, Voyager (Wang et al., 2023a) maintains a curriculum library. Notably, current methods are limited to creating new code skills to interact with external environments.
- **Updating retrieval.** To our knowledge, these learning options are not studied in recent language agents. Retrieval is usually considered a basic action designed with some fixed implementation (e.g., BM25 or dense retrieval), but research in query/document expansion (Nogueira et al., 2019; Wang et al., 2023c; Tang et al., 2023a) or retrieval distillation (Izacard et al., 2021) may be helpful for language agents to learn better retrieval procedures.
- **Updating learning or decision-making.** Finally, it is theoretically possible for CoALA agents to learn new procedures for learning or decision making, thus providing significant adaptability. In general, however, updates to these procedures are risky both for the agent’s functionality and alignment. At present, we are not aware of any language agents that implement this form of learning; we discuss such possibilities more in Section 6.

While RL agents usually fix one way of learning (e.g., Q-learning, PPO, or A3C) and learn by updating model parameters, language agents can select from a diversity of learning procedures. This allows them to learn rapidly by storing task-relevant language (cheaper and quicker than parameter updates), and leverage multiple forms of learning to compound their self-improvement (e.g., Generative Agents discussed in Section 5).

Finally, while our discussion has mostly focused on adding to memory, modifying and deleting (a case of “unlearning”) are understudied in recent language agents. We address these areas more in Section 6.

Learn = Write long-term memory

- Unify various things
- Reveal lots of gaps

Guide Building of (Future) Language Agents

1. Prompt engineering
2. Retrieval corpora
3. In-context learning, finetuning
4. External actions, tools



1. Working memory & reasoning
2. Long-term memory
3. Learning
4. Action space



1. Systematic prompting & parsing (e.g. OpenAI function calls, Guidance, ...)
2. Writing to memory, beyond just retrieval (e.g. Reflexion, Generative Agents, Voyager...)
3. Various learning understudied (meta-prompt, update code, unlearn, mixed learning)
4. Synergizing effects & safety analysis via action space



CoALA:

- Agent = LLM + Memory + Code
 - Memory = episodic / semantic / procedural (stores LLM+code)
 - Code = decision / action procedures
 - Action = grounding / reasoning / learning / retrieval
-
- New techniques can fix old issues
 - Old insights can guide new development

Read the paper for **much more** information!

Part 2. Evaluation

Evaluation of Language Models

- | | | |
|-----------|--|--|
| 0. | easy to solve, easy to evaluate | (Most existing NLP tasks are becoming..) |
| 1. | easy to solve, hard to evaluate | (Open-ended Text Generation) |
| 2. | hard to solve, easy to evaluate | (Games; Coding?) |
| 3. | hard to solve, hard to evaluate | (Most recent agent tasks?) |

Easy to evaluate: cheap, fast, consistent, (not too subjective)

Evaluation: how to be cheap, fast, yet high-quality?



SH please evaluate the code and give a score



Sure, I can provide an evaluation of the code based on a scale of 1 to 10 to rate each aspect. Keep in mind that this is a subjective evaluation with many possible considerations. Here's the evaluation:

1. **Readability (9/10):** The code is well-structured and easy to follow. However, it could benefit from some additional comments to clarify the purpose of the methods and the logic behind the functions.
2. **Efficiency (7/10):** The iterative method is efficient and works well for the given data.

$$r = r_{type} \cdot \frac{|U_{att} \cap Y_{att}| + 1}{|U_{att}| + |Y_{att}| + 1}$$
$$r_{type} \in \{0, 0.1, 0.5, 1\}$$

Human Evaluation



High-quality (esp. if you're OpenAI): RLHF



Not Scalable (if you're not OpenAI): expensive and slow to collect data

LM Evaluation



Not High-quality (yet): hallucinations



Scalable (somewhat): unlimited, but with costs

“Rule-based” Evaluation



High-quality: if we leverage domain priors!



Scalable: unlimited, free, fast

Evaluation of Language Models

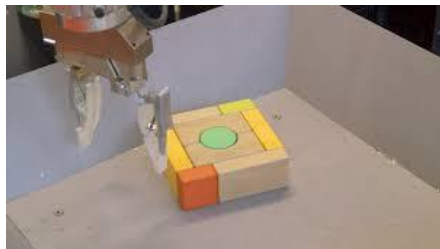
- | | | |
|-----------|--|--|
| 0. | easy to solve, easy to evaluate | (Most existing NLP tasks are becoming..) |
| 1. | easy to solve, hard to evaluate | (Open-ended Text Generation) |
| 2. | hard to solve, easy to evaluate | (Games; Coding?) |
| 3. | hard to solve, hard to evaluate | (Most recent agent tasks?) |

Easy to evaluate: cheap, fast, consistent, (not too subjective)

Evaluation of Language Agents

1. **Collie**: make text generation hard to solve and easy to evaluate!
2. **InterCode**: make coding interactive
3. **WebShop**: make web tasks easy to evaluate

Environment: how to be cheap, fast, yet useful?



Physical World / Humans

😊 **Practical:** robots / chatbots

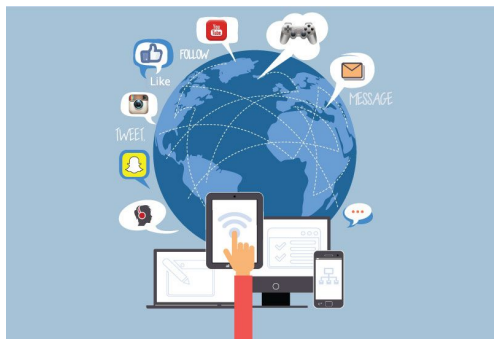
🙄 **Not Scalable:** expensive and slow to collect data



Digital Simulations / Games

🥵 **Not Practical:** sim-to-real is hard

 **Scalable:** free, unlimited interactions






Digital Applications (Internet, code, software, ...)

😊 **Practical:** important tasks to automate

😊 **Scalable:** huge scale, rich complexity, free and fast

Overview

	1. WebShop 	2. InterCode 	3. Collie 
Task	“Find a t-shirt that...”	“Remove the file that...”	“Write a paragraph that...”
Action	Web actions	Code as action	Text as action
Observation feedback	Webpage	Code execution result	Synthetic constraint satisfaction feedback
Reward feedback	Attribute rule-based	Unit test or file diff	Grammar rule-based

COLLIE: Systematic Construction of Constrained Text Generation Tasks

Shunyu Yao* Howard Chen* Austin W. Hanjie* Runzhe Yang* Karthik Narasimhan

Department of Computer Science, Princeton University

{shunyuy, hc22, hjwang, runzhey, karthikn}@princeton.edu

Constrained Text Generation

- A traditional and important NLP (seq2seq) task: constraints -> text
- **Prior benchmarks:** fixed constraint type and too simple for LLMs
 - “Generate a sentence with dog, catch, happy.”
- **Collie’s goal:**
 - Diverse and arbitrarily hard constraints for LLMs, yet guaranteed to be solvable
 - Automatic task construction + evaluation, without human efforts!
 - Challenge language understanding/generation, semantic planning, logical/arithmetic reasoning, ...
- **Core idea:** leverage the infinite expressivity of **grammar**

Collie: Grammar

$S \rightarrow (\text{level}(\xi) = \ell) \wedge M$ (constraint specification) (1)

$M \rightarrow C \mid C \wedge M \mid C \vee M$ (multi-constraint) (2)

$C \rightarrow \text{count}(T, \ell, v_{\text{str}} \mid \ell') \oplus v_{\text{num}} \mid \text{pos}(T, \ell, v_{\text{num}}) \circ v_{\text{str}}$ (base-constraint) (3)

$T \rightarrow \xi \mid \text{pos}(T, \ell, v_{\text{num}})$ (text) (4)

$\ell \rightarrow \text{char} \mid \text{word} \mid \text{sentence} \mid \text{paragraph} \mid \text{passage}$ (level) (5)

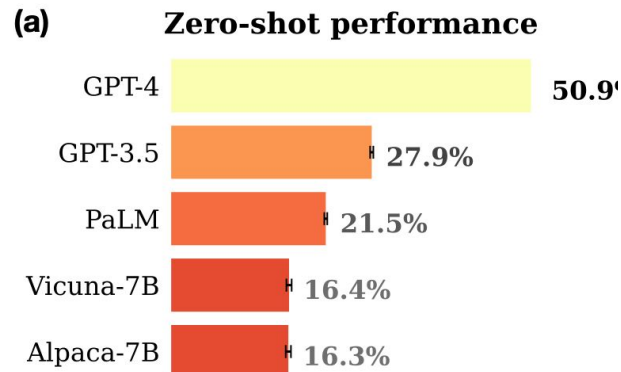
$\circ \rightarrow = \mid \neq \quad \oplus \rightarrow = \mid \neq \mid > \mid < \mid \leq \mid \geq$ (relation) (6)

$v_{\text{str}} \in \Sigma^* \quad v_{\text{num}} \in \mathbb{Z}$ (value) (7)

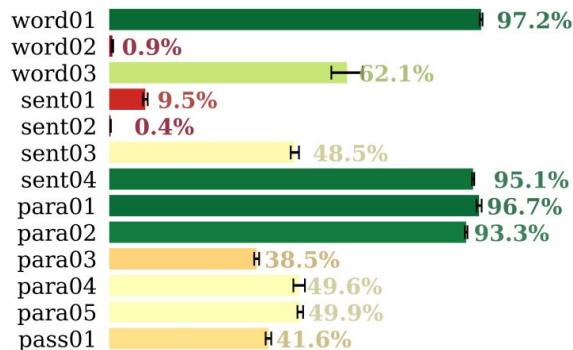
- **Few core concepts:** count, pos, level
- But compositionality yields power
- Easily extensible (e.g. POS, sentiment, topic, ...)

Collie-v1: 2,080 Constraints across 13 Types

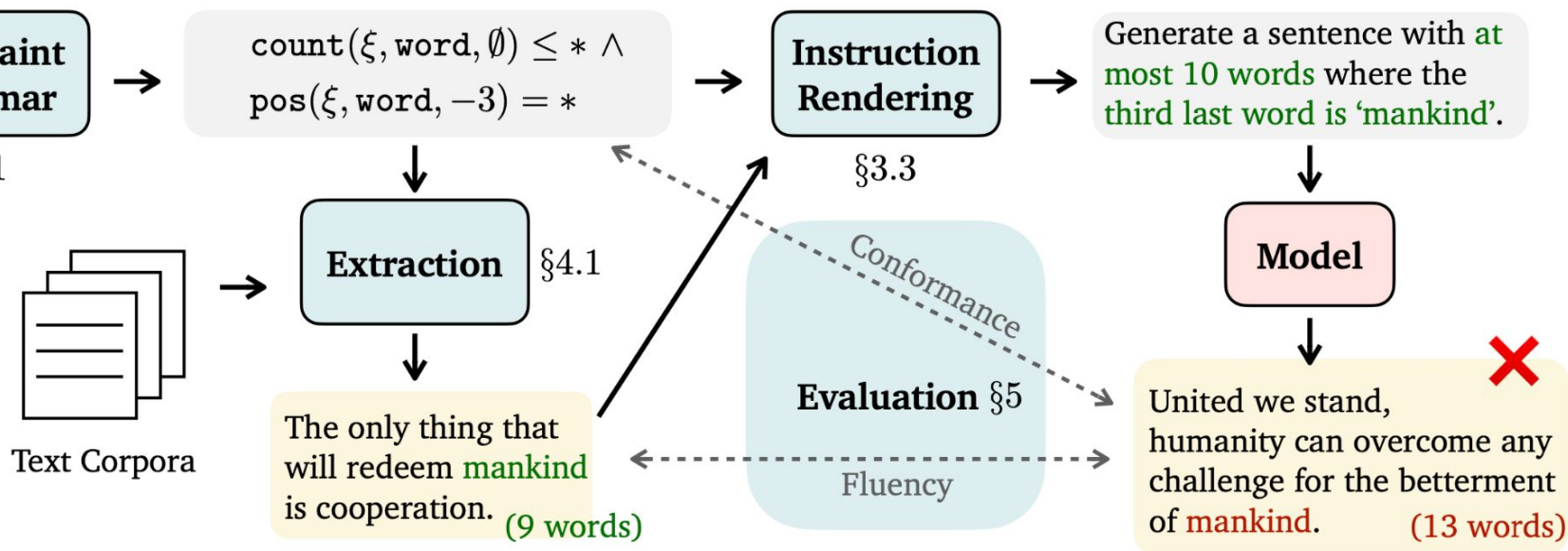
ID	Example instruction	Multi-constraint M
word01	Generate a word with at least 15 letters.	$\text{count}(\xi, \text{char}, \text{word}) \geq 15$
word02	Generate a word with 10 letters, where letter 1 is 's', letter 3 is 'r', letter 9 is 'e'.	$\text{count}(\xi, \text{char}, \text{word}) = 10 \wedge \text{pos}(\xi, \text{char}, 1) = \text{'s'} \wedge \text{pos}(\xi, \text{char}, 3) = \text{'r'} \wedge \text{pos}(\xi, \text{char}, 9) = \text{'e'}$
word03	Generate a word with at most 10 letters and ends with "r".	$\text{count}(\xi, \text{char}, \text{word}) \leq 10 \wedge \text{pos}(\xi, \text{char}, -1) = \text{'r'}$
sent01	Please generate a sentence with exactly 82 characters. Include whitespace into your character count.	$\text{count}(\xi, \text{char}, \text{sentence}) = 82$
sent02	Generate a sentence with 10 words, where word 3 is "soft" and word 7 is "beach" and word 10 is "math".	$\text{count}(\xi, \text{word}, \text{sentence}) = 10 \wedge \text{pos}(\xi, \text{word}, 3) = \text{"soft"} \wedge \text{pos}(\xi, \text{word}, 7) = \text{"beach"} \wedge \text{pos}(\xi, \text{word}, 10) = \text{"math"}$
sent03	Generate a sentence with at least 20 words, and each word less than six characters.	$\text{count}(\xi, \text{word}, \text{sentence}) \geq 20 \wedge \text{count}(\xi, \text{char}, \text{word}) \leq 6$
sent04	Generate a sentence but be sure to include the words "soft", "beach" and "math".	$\text{count}(\xi, \text{word}, \text{'soft'}) > 0 \wedge \text{count}(\xi, \text{word}, \text{'beach'}) > 0 \wedge \text{count}(\xi, \text{word}, \text{'math'}) > 0$
para01	Generate a paragraph where each sentence begins with the word "soft".	$\text{pos}(\text{pos}(\xi, \text{sentence}, 1), \text{word}, 1) = \text{'soft'} \wedge \text{pos}(\text{pos}(\xi, \text{sentence}, 2), \text{word}, 1) = \text{'soft'} \wedge \dots$
para02	Generate a paragraph with at least 4 sentences, but do not use the words "the", "and" or "of".	$\text{count}(\xi, \text{sentence}, \text{paragraph}) \geq 4 \wedge \text{count}(\xi, \text{word}, \text{'the'}) = 0 \wedge \text{count}(\xi, \text{word}, \text{'and'}) = 0 \wedge \text{count}(\xi, \text{word}, \text{'of'}) = 0$
para03	Generate a paragraph with exactly 4 sentences, each with between 10 and 15 words.	$\text{count}(\xi, \text{sentence}, \text{paragraph}) = 4 \wedge \text{count}(\xi, \text{word}, \text{sentence}) \geq 10 \wedge \text{count}(\xi, \text{word}, \text{sentence}) \leq 15$
para04	Generate a paragraph with at least 3 sentences, each with at least 15 words.	$\text{count}(\xi, \text{sentence}, \text{paragraph}) \geq 3 \wedge \text{count}(\xi, \text{word}, \text{sentence}) \geq 15$
para05	Generate a paragraph with 2 sentences that end in "math" and "rock" respectively.	$\text{count}(\xi, \text{sentence}, \text{paragraph}) = 2 \wedge \text{pos}(\text{pos}(\xi, \text{sentence}, 1), \text{word}, -1) = \text{"math"} \wedge \text{pos}(\text{pos}(\xi, \text{sentence}, 2), \text{word}, -1) = \text{"rock"}$
pass01	Generate a passage with 2 paragraphs, each ending in "I sit." and "I cry." respectively.	$\text{count}(\xi, \text{paragraph}, \text{passage}) = 2 \wedge \text{pos}(\text{pos}(\xi, \text{paragraph}, 1), \text{sentence}, -1) = \text{"I sit."} \wedge \text{pos}(\text{pos}(\xi, \text{paragraph}, 2), \text{sentence}, -1) = \text{"I cry."}$



Constraint satisfaction rate - GPT-4



Task Construction is Fully Automatic



- Human just specify constraint types
- Collie automatically extracts constraint “values” from corpora
- Rule-based instruction rendering and text evaluation (extensible)

Feedback helps!

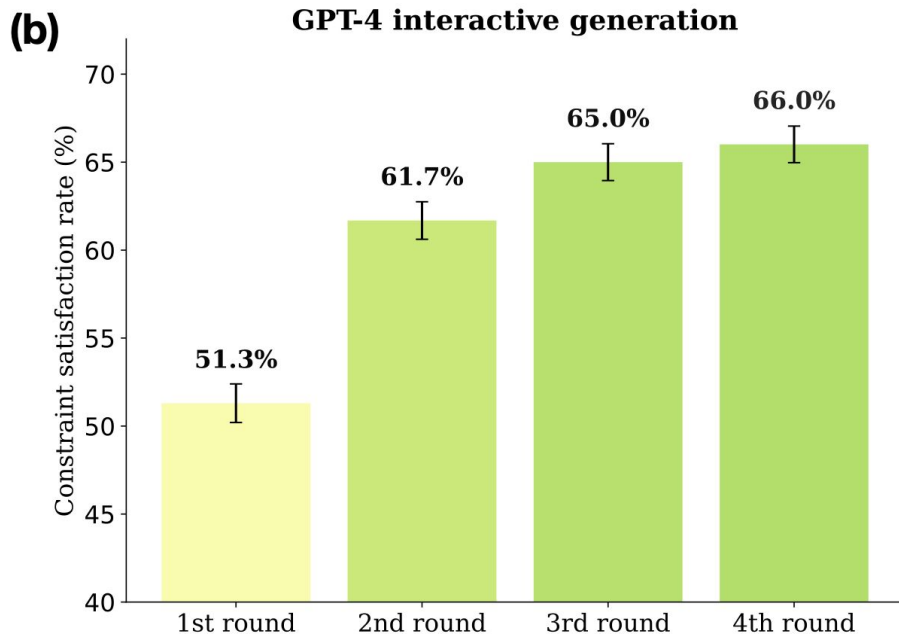
SH generate a word with 10 letters, where letter 1 is "s", letter 3 is "r", letter 9 is "e"

Sure, here's a word that meets your criteria: "strenuous".

SH your word only has 9 letters, and letter 9 is "s", not "e".

I apologize for the mistake. Here's a correct word that meets your criteria: "strategies".

Many other cool findings in the paper!



InterCode: Standardizing and Benchmarking Interactive Coding with Execution Feedback

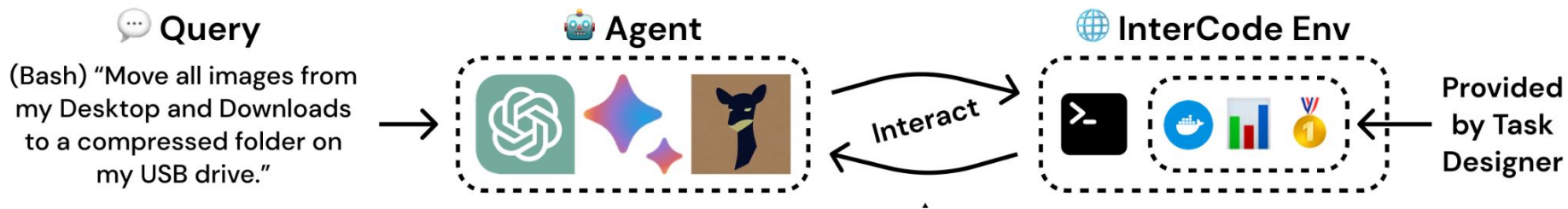
John Yang* Akshara Prabhakar* Karthik Narasimhan Shunyu Yao

Department of Computer Science, Princeton University

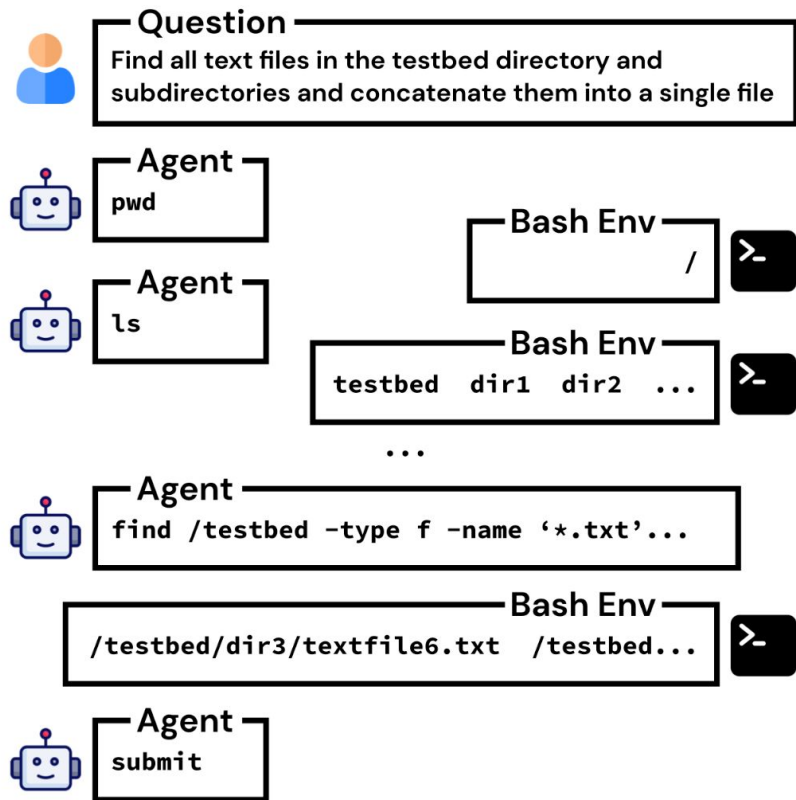
{jy1682, ap5697, karthikn, shunyuy}@princeton.edu

Code Interaction

- Static NL2Code benchmarks: HumanEval, Spider, NL2Bash, ...
- But humans code in a fundamentally interactive manner!
- Some interactive/execution-based methods, but no standard benchmark

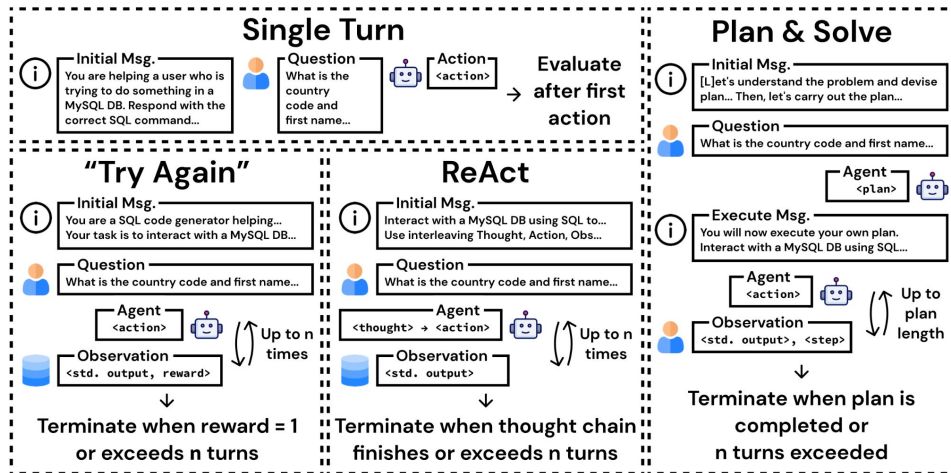


InterCode Setup



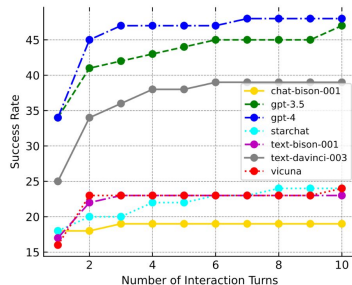
- Standard RL env:
 - **Environment:** Docker-based Python/SQL/bash terminals
 - **Action:** code command
 - **Observation:** execution result
- Benefits
 - Safe and reproducible
 - Unlock new tasks (e.g. CTF)
 - Unlock new evaluations (e.g. Bash)
 - Unlock new methods (e.g. Plan-and-solve)

InterCode: new methods

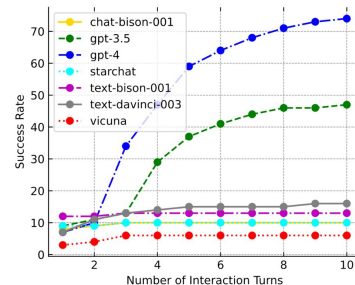


	Try Again ($n = 10$)			ReAct ($n = 10$)			Plan & Solve		
	SR	Turns	Error %	SR	Turns	Error %	SR	Turns	Error %
SQL	47.3	7.25	46.4	58.7	5.30	6.94	49.1	4.29	16.2
Bash	46.5	6.15	24.9	20.5	4.40	20.4	28.0	6.65	53.3

- Interactive >> seq2seq
- Different interactive methods have different tradeoffs
- Large room for improvement



(a) Success rate vs. turns for InterCode-Bash



(b) Success rate vs. turns for InterCode-SQL

Future: Coding -> Software engineering?



SWE-bench: Can Language Models Resolve Real-World GitHub Issues?

Carlos E. Jimenez^{* 1,2} John Yang^{* 1,2} Alexander Wettig^{1,2}
 Shunyu Yao^{1,2} Kexin Pei³ Ofir Press^{1,2} Karthik Narasimhan^{1,2}
¹Princeton University ²Princeton Language and Intelligence ³University of Chicago

Model	% Resolved	% Apply
ChatGPT-3.5	0.50	8.40
Claude 2	3.60	38.10
GPT-4*	1.30	10.00
SWE-Llama 7b	3.00	54.80
SWE-Llama 13b	4.00	52.10

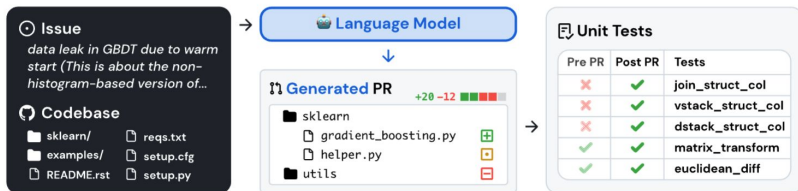


Figure 1: SWE-bench sources task instances from real-world Python repositories by connecting GitHub issues to merged pull request solutions that resolve related tests. Provided with the issue text and a codebase snapshot, models generate a patch that is evaluated against real tests.

WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents

Shunyu Yao* Howard Chen* John Yang Karthik Narasimhan

Department of Computer Science, Princeton University

{shunyuy, howardchen, jy1682, karthikn}@princeton.edu

Web Interaction

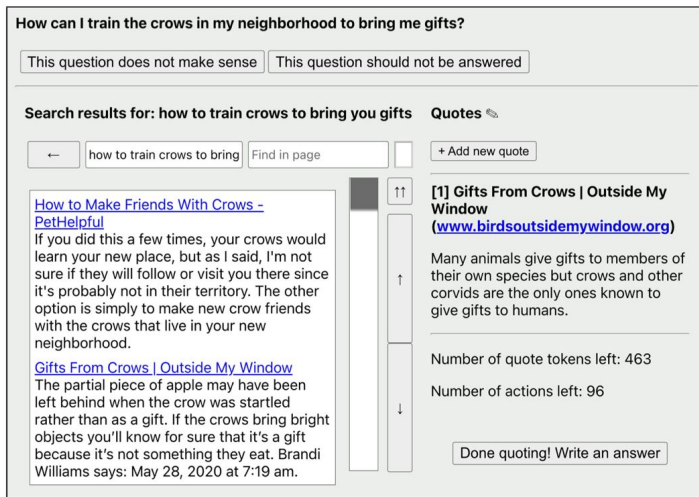
Table 3

Query

For the last 8
house arrest

In the winter
at Rainier Pa

This compar
had a guaran



(a) Screenshot from the demonstration interface.

◆Question
How can I train the crows in my neighborhood to bring me gifts?

◆Quotes
From Gifts From Crows | Outside My Window (www.birdsoutsidemymwindow.org)
> Many animals give gifts to members of their own species but crows and other corvids are the only ones known to give gifts to humans.

◆Past actions
Search how to train crows to bring you gifts
Click Gifts From Crows | Outside My Window www.birdsoutsidemymwindow.org
Quote
Back

◆Title
Search results for: how to train crows to bring you gifts

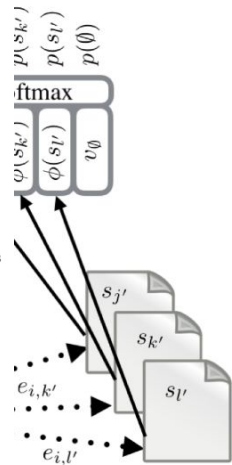
◆Scrollbar: 0 - 11

◆Text
[0]How to Make Friends With Crows - PetHelpfulIf you did this a few times, your crows would learn your new place, but as I said, I'm not sure if they will follow or visit you there since it's probably not in their territory. The other option is simply to make new crow friends with the crows that live in your new neighborhood.

[1]Gifts From Crows | Outside My Windowwww.birdsoutsidemymwindow.org
The partial piece of apple may have been left behind when the crow was startled rather than as a gift. If the crows bring bright objects you'll know for sure that it's a gift because it's not something they eat. Brandi Williams says: May 28, 2020 at 7:19 am.

◆Actions left: 96
◆Next action

(b) Corresponding text given to the model.



Nogueira et al. End-to-End Goal-Driven Web Navigation
Nakano et al. WebGPT: Browser-assisted question-answering with human feedback

Mini WWikiNav:WebGPT: RLHFe games"mes"

WebShop

A

WebShop

search

Instruction:
i'm looking for a small portable folding desk that is already fully assembled; it should have a khaki wood finish, and price lower than 140.00 dollars

portable folding desk khaki wood **1** Search

Description:Product laptop desk.Product walnut.Product weight: 4.6pounds.Material: high quality thick steel pipe, black brushed smooth table top, increase the length and width of the table, it is possible to place the computer and various items.Function: Can be used as computer desk, dining table, bedside table.Product size: 23.6x15.7x11 inches

item-detail

- **[Large Size]** styling with light wood finish. Holds laptops up to 17 inches. It also have spacious space (23.6x15.7x11 inches) for your laptop, notebook, mouse, pen and coffee. Its generous size gives this versatile desk even more flexibility.
- **[Wide Application]** Our foldable lap desk can be used as a


item-detail

Back to Search

Page 1 (Total results: 50)

Next >


2 results



B09GCB186B


MENHG Folding Breakfast Tray Table, Efficient Home Laptop Notebook Computer Desk, Portable Writing Study Desk, Sturdy Home Office Table Workstation

\$109.0



B08PZSC2W8

KPSP Folding Study Desk Bed Breakfast Serving Tray Table Efficient Home Laptop Notebook Computer Desk Portable Standing Desk for Small Space Bedroom



item

MENHG Folding Laptop Table Bed Desk PC Lap Desk with Drawer Book Stand Reading Holder Leg Space Laptop Bed Tray Foldable Lazy Table Breakfast Desk Sofa Small Desk for Small Space

Price: \$100.0

Rating: N/A

Description Overview

Color black khaki white **3**

Buy Now 5

Reward: 1.0

Why Shopping?

- Scalable **environment**: rich dynamics, scalable items
- Scalable **task**: well-defined problem, possible for automatic reward synthesis
- Interesting **challenges**: multi-modal understanding, decision making, etc.

Construction

Orange: hidden from agents!

- Scrape 1.18M products from amazon.com, text mine attributes



Department

Electronics

Accessories & Supplies

Camera & Photo

Car & Vehicle Electronics

Cell Phones & Accessories

Computers & Accessories

Electronics Warranties

GPS, Finders & Accessories

Headphones

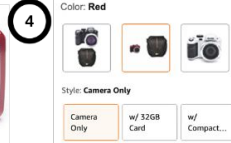
Home Audio

Office Electronics



1 KODAK PIXPRO Astro Zoom AZ421-BK 16MP Digital Camera with 42X Optical Zoom and 3" LCD Screen (Black)

2 \$174.95



5 About this item

- Powerful 16.1-Megapixel CCD sensor gives you room to enlarge, zoom and crop to content without losing out on life-like clarity or quality.
- 42x optical zoom and a 24mm wide angle lens Moves you closer to the subject and every frame; 1080P
- Face/Cat/dog detection detects facial features and enhances it for true subject high your four-legged Cat and dog friends. Blink/smile detection helps you to capture the moment.
- Face Beautifier mode and loads of built-in touch-up features allow you to reduces f enhance skin tone and the eyes of your portrait subjects instantly.
- Use the Panorama mode to create a seamless panoramic picture to 180-degree eas the camera across your subject. Continuous shooting
- The a2421 takes a standard SD/SDHC memory card that is at least Class 4 and no li 32GB

6 Product Description

Passion and performance Go hand in hand when with our Kodak PIXPRO AZ421 digital cam optical image stabilization delivers crisp, clear 16 Megapixel close-ups, panorama or HD vide powerful yet user-friendly settings make photography easy, fun and zero hassle. Kodak PIXP


7 Optical zoom
LCD screen
Memory card
Digital camera

Construction

Orange: hidden from agents!

- Scrape 1.18M products from amazon.com, text mine attributes
- Build synthetic website with aligned text interface


Next >



B09Q3B186B

MENHG Folding Breakfast Tray Table, Efficient Home Laptop Notebook Computer Desk, Portable Writing Study Desk, Sturdy Home Office Table Workstation

\$109.0



B09P5ZBCWR

KPSP Folding Study Desk Bed Breakfast Serving Tray Table Efficient Home Laptop Notebook Computer Desk Portable Standing Desk for Small Space Bedroom

HTML mode

B

Instruction:
I'm looking for a small portable folding desk that is already fully assembled [...]
[btn] Back to Search [/btn]
Page 1 (Total results: 50) [btn] Next [/btn]
[btn] MENHG Folding Breakfast Tray [...] [/btn]
\$109.0
[btn] KPSP Folding Study Desk Bed [...] [/btn]

Simple mode

Construction

Orange: hidden from agents!

- Scrape 1.18M products from amazon.com, text mine attributes
- Build synthetic website with aligned text interface
- **Human** instructions & **Automatic** reward via product attributes/options/prices/types

Goal product (U)



Fujifilm X-T1 16 MP Mirrorless Digital Camera with 3.0-Inch LCD (Body Only) (Graphite Silver & Weather Resistant) (Renewed)

Price: \$904.95

Attributes

certified refurbished
water resistant
high performance

color

silver

graphite silver

configuration

base

international version

Human

Goal Instruction

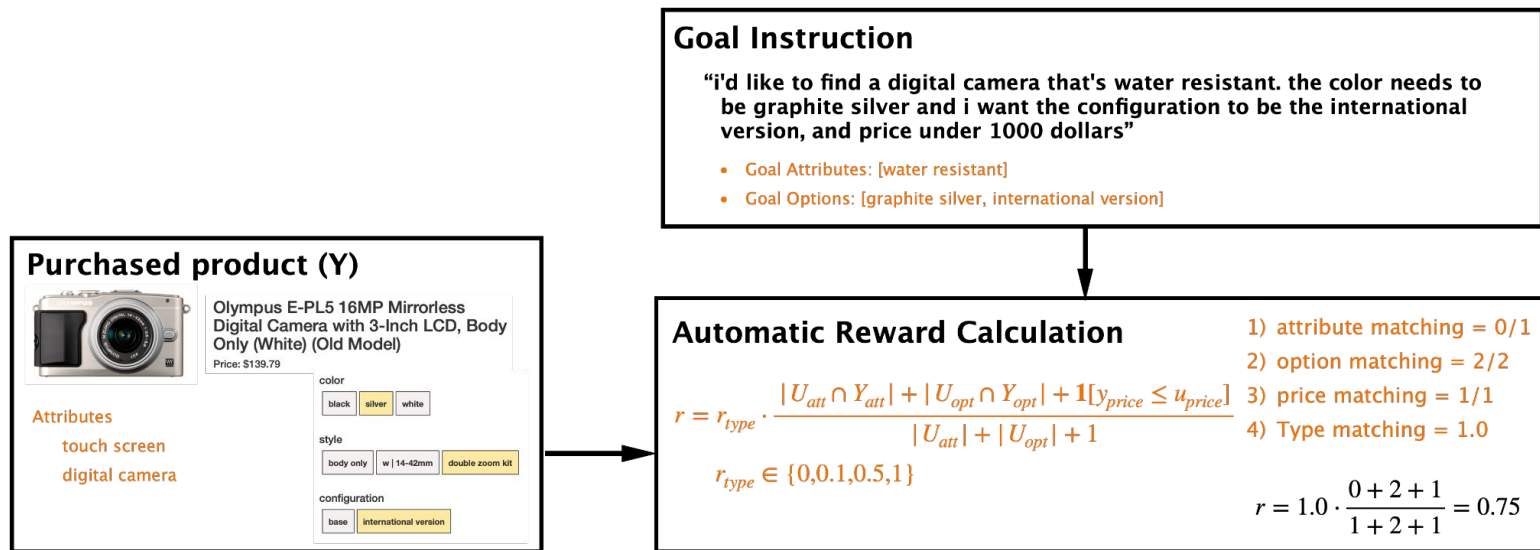
“i'd like to find a digital camera that's water resistant. the color needs to be graphite silver and i want the configuration to be the international version, and price under 1000 dollars”

- Goal Attributes: [water resistant]
- Goal Options: [graphite silver, international version]

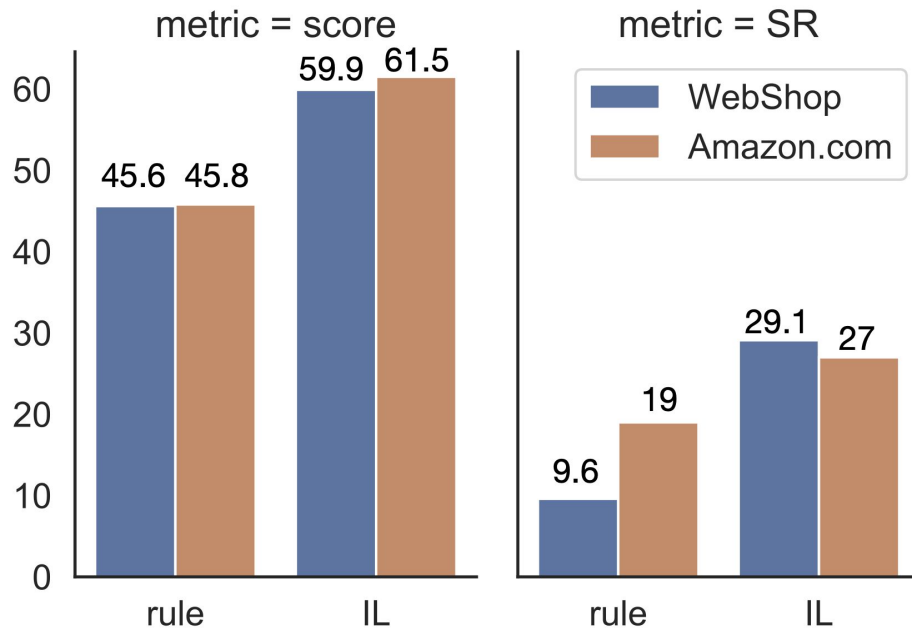
Construction

Orange: hidden from agents!

- Scrape 1.18M products from amazon.com, text mine attributes
- Build synthetic website with aligned text interface
- **Human** instructions & **Automatic** reward via product attributes/options/prices/types



Sim-to-real transfer (Amazon/eBay)

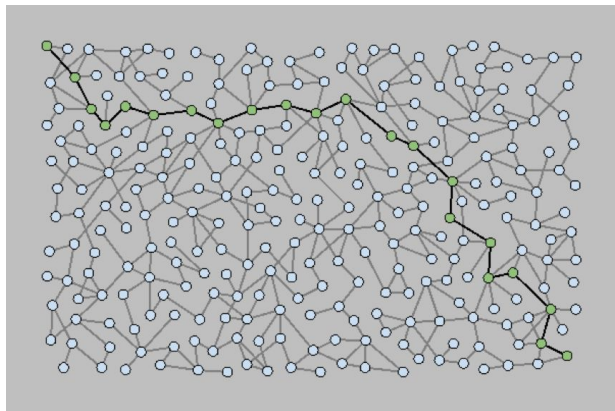


Learned policy generalizes to different search engine and products

WebShop: Summary

- First large-scale, realistic web interaction benchmark
 - Trending in 2023: WebArena, Mind2Web, ...
- **Synthetic website** provides controlled development, transfers to real websites
- **Task priors** (self/model-supervision) provides scalable reward

(Individual) Human reward finetuning on top of synthetic reward pre-training?



Text mining

touch screen
digital camera



Sentiment analysis

Users find it good..

Summarization

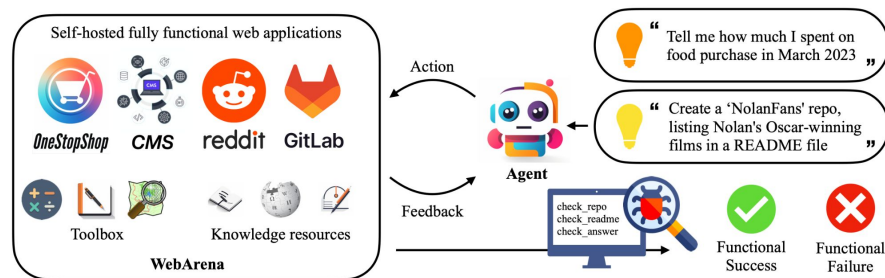
This is a digital...

Image Detection

Optical zoom...

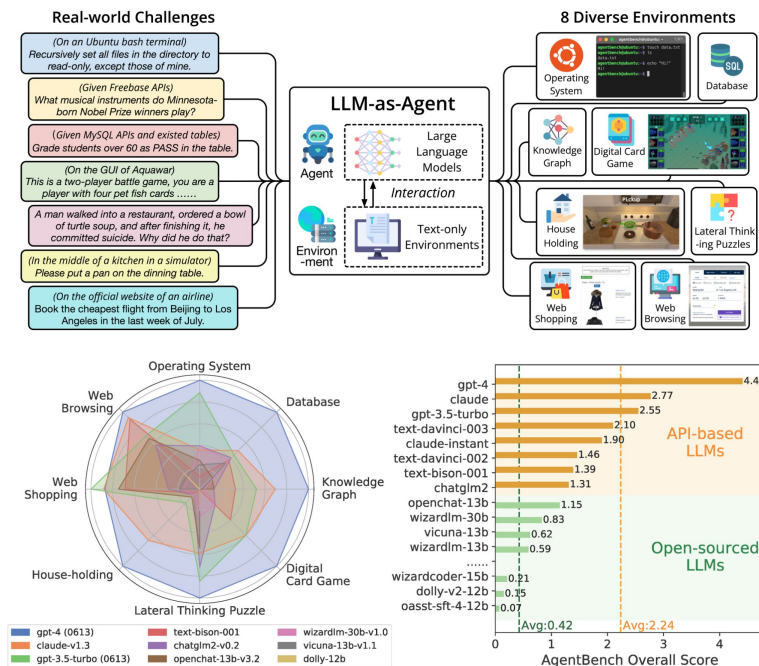
Other Recent Benchmarks

WebArena



Function	ID	Intent	Eval Implementation
$r_{\text{info}}(a^*, \hat{a})$	1	Tell me the name of the customer who has the most cancellations in the history	<code>exact_match(\hat{a}, "Samantha Jones")</code>
	2	Find the customer name and email with phone number 8015551212	<code>must_include(\hat{a}, "Sean Miller")</code> <code>must_include(\hat{a}, "sean@gmail.com")</code>
	3	Compare walking and driving time from AMC Waterfront to Randyland	<code>fuzzy_match(\hat{a}, "Walking: 2h58min")</code> <code>fuzzy_match(\hat{a}, "Driving: 21min")</code>
$r_{\text{prog}}(s)$	4	Checkout merge requests assigned to me	<code>url = locate_last_url(s)</code> <code>exact_match(URL, "gitlab.com/merge_requests?assignee_username=byteblaze")</code>
	5	Post to ask "whether I need a car in NYC"	<code>url = locate_latest_post_url(s)</code> <code>body = locate_latest_post_body(s)</code> <code>must_include(URL, "/f/nyc")</code> <code>must_include(body, "whether I need a car in NYC")</code>

AgentBench



Summary

- Language agents are a new & different kind of agents that rely on LLM reasoning
- We have a lot of ideas (and hypes), but we lack theories and benchmarks
 - To formulate language agents, use classical insights from AI and CogSci
 - To evaluate language agents, use real-world interactive tasks + “good” metrics
 - **Where academia could uniquely help**
- Future directions for language agents.....
 - Check section 6 of the CoALA paper
 - Chat with me (in the afternoon or email)
 - <https://tinyurl.com/shunyu-feedback>

Thanks!