

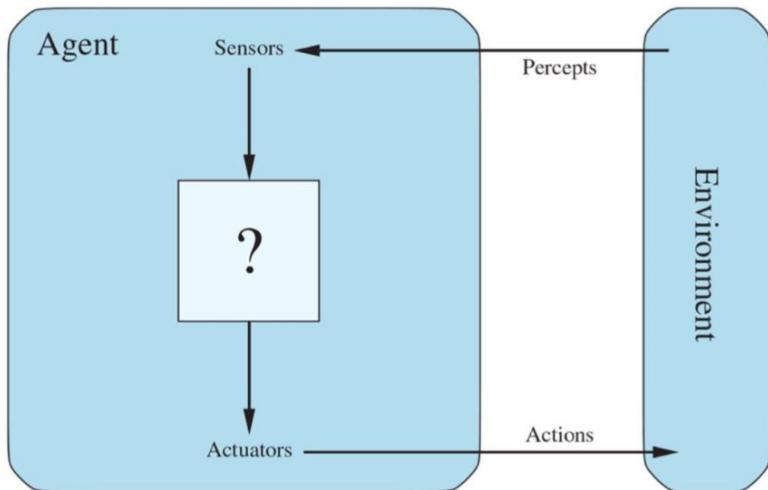


CS 957, System-2 AI Language Agents

Mohammad Hossein Rohban | May. 2025

Sharif University of Technology

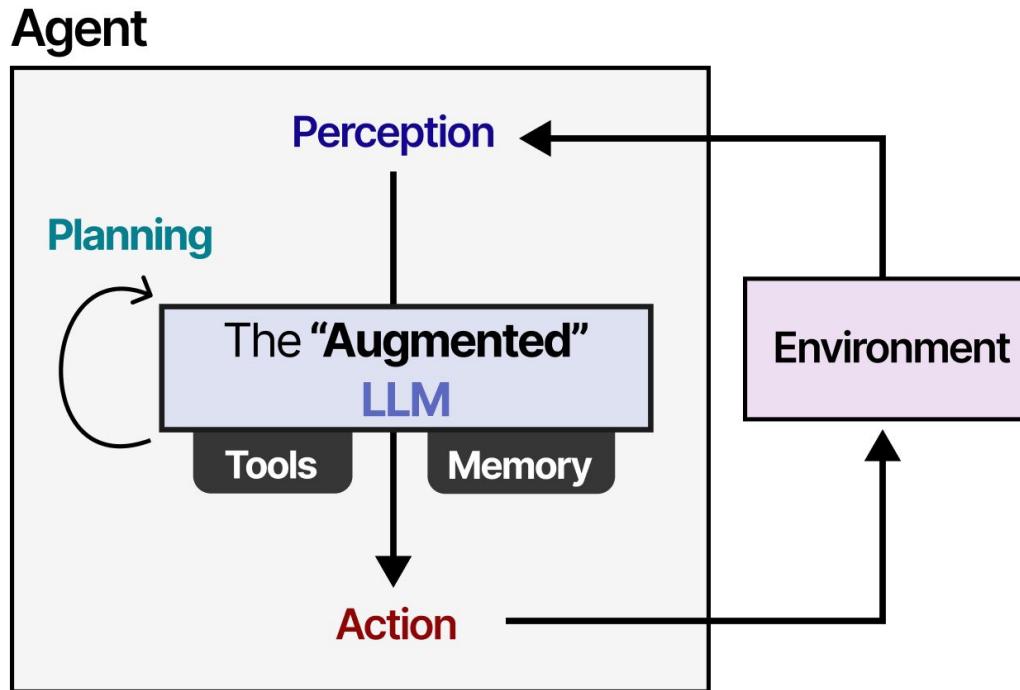
Agents?



"An **agent** is anything that can be viewed as perceiving its **environment** through **sensors** and acting upon that environment through **actuators**"

— Russel & Norvig, *AI: A Modern Approach*

Agents?



Two Competing Views

LLM-first view We make an LLM into an agent!

- *Implications:* scaffold on top of LLMs, prompting-focused, heavy on engineering

Agent-first view We integrate LLMs into AI agents so they can use language for reasoning and communication!

- *Implications:* All the same challenges faced by previous AI agents (e.g., perception, reasoning, world models, planning) still remain, but we need to **re-examine them through the new lens of LLMs** and tackle new ones (e.g., synthetic data, self-reflection, internalized search)

Design Philosophy

LLM-first:

- Agents serving as "wrappers" that **augment** the model with tools (e.g., calculators, APIs) and memory systems.
- overcoming LLM limitations like **statelessness** and **knowledge** cutoffs.
- e.g. ChatGPT with **plugins** (e.g., browsing the web or generating code) uses the LLM as the "brain" and tools as extensions.

Agent-first:

- Starts with the agent as a **holistic** system
- LLM is **one component among many** (e.g., memory, planning modules, tools).
- Agent is designed to act **autonomously** in dynamic environments.
- e.g. A financial analysis agent that autonomously retrieves real-time stock data via APIs, **analyzes trends**, and **generates reports** without user intervention.

Autonomy and Planning

LLM-first:

- Limited autonomy: actions are constrained by **predefined workflows** or user prompts.
- e.g. A coding assistant suggesting code snippets but requiring user approval to execute them.

Agent-first:

- Agents **plan and execute** multi-step tasks **independently**, using frameworks like **ReAct** or **LangGraph** for dynamic problem-solving.
- e.g. A research agent decomposing a complex query into **sub-questions**, retrieving data, and synthesizing answers without human input.

REACT: SYNERGIZING REASONING AND ACTING IN LANGUAGE MODELS

Shunyu Yao^{*,1}, Jeffrey Zhao², Dian Yu², Nan Du², Izhak Shafran², Karthik Narasimhan¹, Yuan Cao²

¹Department of Computer Science, Princeton University

²Google Research, Brain team

¹{shunyuy,karthikn}@princeton.edu

²{jeffreyzhao,dianyu,dunan,izhak,yuancao}@google.com

Reasoning and Acting

The LLM abilities for **reasoning** (e.g. chain-of-thought prompting) and **acting** (e.g. action-plan generation) have primarily been studied as **separate** topics.

There is a **synergy** between the two. How?

(1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

(1a) Standard

Answer: iPod



(1c) Act-Only

Act 1: **Search**[Apple Remote]

Obs 1: The Apple Remote is a remote control ...

Act 2: **Search**[Front Row]

Obs 2: Could not find [Front Row]. Similar: ...

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

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

Act 4: **Finish**[yes]



(1b) CoT (Reason Only)

Thought: Let's think step by step. Apple Remote was originally designed to interact with Apple TV.

Apple TV can be controlled by iPhone, iPad, and iPod

Touch. So the answer is iPhone, iPad, and iPod Touch.

Answer: iPhone, iPad, iPod Touch



(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]



(2) AlfWorld

You are in the middle of a room. Looking quickly around you, you see a cabinet 6, a cabinet 1, a coffee machine 1, a countertop 3, a stove burner 1, and a toaster 1.

Your task is to: Put some pepper shaker on a drawer.

(2a) Act-Only

Act 1: Go to drawer 1

Obs 1: The drawer 1 is closed.

Act 2: Open drawer 1

Obs 2: You open the drawer 1. The drawer 1 is open. In it, you see a dishponge 2, and a spoon 1.

Act 3: Go to sinkbasin 1

Obs 3: On the sinkbasin 1, you see a dishponge 3, a spatula 1, and a spoon 2.

Act 4: Take peppershaker 1 from sinkbasin 1

Obs 4: Nothing happens.

Act 5: Take peppershaker 1 from sinkbasin 1

Obs 5: Nothing happens.



(2b) ReAct (Reason + Act)

Act 1: **Think**[First I need to find a pepper shaker...more likely to appear in cabinets (1-6), countertops (1-3), ...]

Act 2: Go to cabinet 1

Obs 2: On the cabinet 1, you see a vase 2.

(Here the agent go to cabinet 1, then cabinet 2, 3, then countertop 1 and 2)

Act 6: Go to countertop 3

Obs 6: On the countertop 3, you see a apple 1, a bread 1, a pepper shaker 1, and a vase 3.

Act 7: Take pepper shaker 1 from countertop 3

Obs 7: You pick up the pepper shaker 1 from the countertop 3.

Act 8: **Think**[Now I find a pepper shaker 1. Next, I need to put it in/on drawer 1.]

Act 9: Go to drawer 1

Obs 9: Drawer 1 is closed.

Act 10: Open drawer 1

Obs 10: You open Drawer 1 ...

Act 11: Put pepper shaker 1 in/on drawer 1

Obs 11: You put pepper shaker 1 in/on the drawer 1.



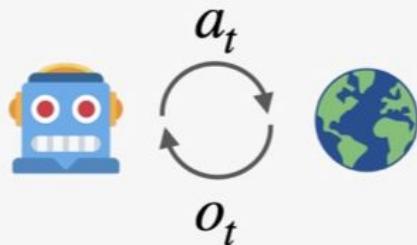
ReAct

Augment the action space with language: $\hat{\mathcal{A}} = \mathcal{A} \cup \mathcal{L}$

$a_t \in L$ (also called a thought) does not affect the external environment → leading to no observation feedback.

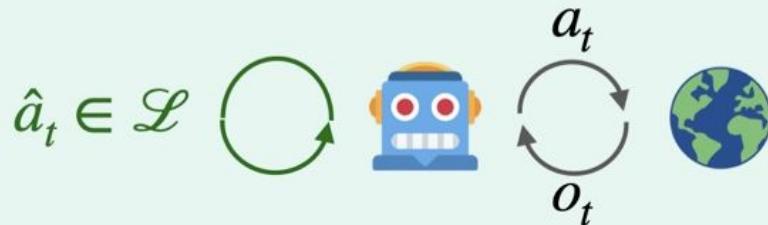
Instead, it composes useful information by reasoning over the current context c_t , and update the context $c_{t+1} = (c_t, a_t)$ to support future reasoning or acting.

Traditional agents: action space A defined by the environment



- External feedback o_t
- Agent context $c_t = (o_1, a_1, o_2, a_2, \dots, o_t)$
- Agent action $a_t \sim \pi(a | c_t) \in A$

ReAct: action space $\hat{A} = A \cup \mathcal{L}$ augmented by reasoning



- $\hat{a}_t \in \mathcal{L}$ can be any language sequence
- Agent context $c_{t+1} = (c_t, \hat{a}_t, a_t, o_{t+1})$
- $\hat{a}_t \in \mathcal{L}$ only updates **internal context**

Various Useful Thoughts

Decomposing task goals and create action plans (2b, Act 1; 1d, Thought 1),

Act 1: Think[First I need to find a pepper shaker...more likely to appear in cabinets (1-6), countertops (1-3), ...]

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

Extracting important parts from observations (1d, Thought 2, 4),

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.

Various Useful Thoughts

Track progress and transit action plans (2b, Act 8),

Obs 7: You pick up the pepper shaker 1 from the countertop 3.

Act 8: Think [Now I find a pepper shaker 1. Next, I need to put it in/on drawer 1.]

Handle exceptions and adjust action plans (1d, Thought 3)

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) .

ReAct Prompting

Randomly select k cases from the training set and manually compose ReAct-format trajectories to use as **few-shot exemplars** in the prompt:

ReAct

Question	What is the elevation range for the area that the eastern sector of the Colorado orogeny extends into?
Thought 1	I need to search Colorado orogeny, find the area that the eastern sector of the Colorado orogeny extends into, then find the elevation range of the area.
Action 1	Search[Colorado orogeny]
Observation 1	The Colorado orogeny was an episode of mountain building (an orogeny) in Colorado and surrounding areas.
Thought 2	It does not mention the eastern sector. So I need to look up eastern sector.
Action 2	Lookup[eastern sector]
Observation 2	(Result 1 / 1) The eastern sector extends into the High Plains and is called the Central Plains orogeny.
Thought 3	The eastern sector of Colorado orogeny extends into the High Plains. So I need to search High Plains and find its elevation range.
Action 3	Search[High Plains]
Observation 3	High Plains refers to one of two distinct land regions
Thought 4	I need to instead search High Plains (United States).
Action 4	Search[High Plains (United States)]

Zero-Shot ReAct Prompting

You are an agent that answers questions by using two actions:

- search[query]: Google search the query. You can also use it to calculate math.
- finish[answer]: return the answer

Your generation should have the following format:

Thought: to analyze the observation and inform the action.

Action: your action

Question: If I have seven trillion dollars today, can I buy Apple, Nvidia, and Microsoft? If not, how much more money do I need?

Results

ReAct → CoT-SC: when ReAct fails to return an answer within given steps, back off to CoT-SC.

CoT-SC → ReAct: when the majority answer among n CoT-SC samples occurs less than n/2 times (i.e. internal knowledge might not support the task confidently), back off to ReAct.

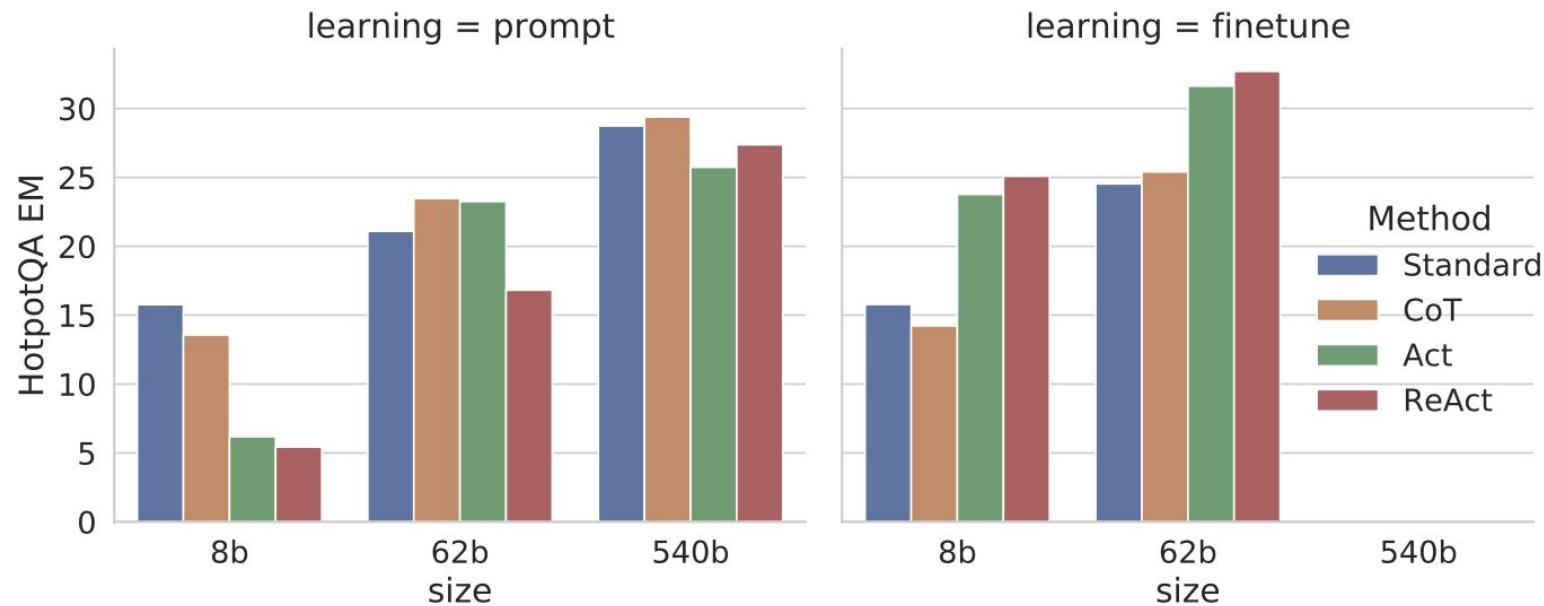
Prompt Method ^a	HotpotQA (EM)	Fever (Acc)
Standard	28.7	57.1
CoT (Wei et al., 2022)	29.4	56.3
CoT-SC (Wang et al., 2022a)	33.4	60.4
Act	25.7	58.9
ReAct	27.4	60.9
CoT-SC → ReAct	34.2	64.6
ReAct → CoT-SC	35.1	62.0
Supervised SoTA^b	67.5	89.5

Table 1: PaLM-540B prompting results on HotpotQA and Fever.

^a HotpotQA EM is 27.1, 28.9, 33.8 for Standard, CoT, CoT-SC in Wang et al. (2022b).

^b (Zhu et al., 2021; Lewis et al., 2020)

Finetuning?? 🤔🤔🤔



In action ...

```
# import libraries
import openai
import os
from langchain.llms import OpenAI
from langchain.agents import load_tools
from langchain.agents import initialize_agent
from dotenv import load_dotenv
load_dotenv()

# load API keys; you will need to obtain these if you haven't yet
os.environ["OPENAI_API_KEY"] = os.getenv("OPENAI_API_KEY")
os.environ["SERPER_API_KEY"] = os.getenv("SERPER_API_KEY")
```

```
llm = OpenAI(model_name="text-davinci-003", temperature=0)
tools = load_tools(["google-serper", "llm-math"], llm=llm)
agent = initialize_agent(tools, llm, agent="zero-shot-react-description", verbose=True)
```

In action ...

```
agent.run("Who is Olivia Wilde's boyfriend? What is his current age raised to the 0.23 power?")
```

```
> Entering new AgentExecutor chain...
I need to find out who Olivia Wilde's boyfriend is and then calculate his age raised to the 0.23
power.
Action: Search
Action Input: "Olivia Wilde boyfriend"
Observation: Olivia Wilde started dating Harry Styles after ending her years-long engagement to
Jason Sudeikis – see their relationship timeline.
Thought: I need to find out Harry Styles' age.
Action: Search
Action Input: "Harry Styles age"
Observation: 29 years
Thought: I need to calculate 29 raised to the 0.23 power.
Action: Calculator
Action Input: 29^0.23
Observation: Answer: 2.169459462491557

Thought: I now know the final answer.
Final Answer: Harry Styles, Olivia Wilde's boyfriend, is 29 years old and his age raised to the
0.23 power is 2.169459462491557.

> Finished chain.
```

Reflexion: Language Agents with Verbal Reinforcement Learning

Noah Shinn

Northeastern University

noahshinn024@gmail.com

Federico Cassano

Northeastern University

cassano.f@northeastern.edu

Ashwin Gopinath

MIT

agopi@mit.edu

Karthik Narasimhan

Princeton University

karthikn@princeton.edu

Shunyu Yao

Princeton University

shunyuy@princeton.edu

What was the catch?

Limited to using **in-context examples** as a way of teaching the agents.

Reinforcement learning with gradient descent require **substantial** amounts of compute and **time**

Reflexion converts binary or **scalar feedback** from the environment **into verbal feedback** in the form of a textual summary

Then added as **additional context** for the LLM agent in the next episode.

	1. Decision making	2. Programming	3. Reasoning
(a) Task	You are in the middle of a room [...] Task: clean some pan and put it in countertop.	Task: You are given a list of two strings [...] of open '(' or close ')' parentheses only [...]	Task: What profession does John Lanchester and Alan Dean Foster have in common?
(b) Trajectory	[...] Action: take pan1 from stoveburner1 Obs: Nothing happens. [...] Action: clean pan1 with sinkbasin1 Obs: Nothing happens. [...]	def match_parens(lst): if s1.count('(') + s2.count('(') == s1.count(')') + s2.count(')'). [...] return 'No'	Think: [...] novelist, journalist, critic [...] novelist, screenwriter [...] common is novelist and screenwriter. Action: "novelist, screenwriter"
(c) Evaluation (internal / external)	Rule/LM Heuristic: Hallucination.	Self-generated unit tests fail: assert match_parens(...)	Environment Binary Reward: 0
(d) Reflection	[...] tried to pick up the pan in stoveburner 1 [...] but the pan was not in stoveburner 1. [...]	[...] wrong because it only checks if the total count of open and close parentheses is equal [...] order of the parentheses [...]	[...] failed because I incorrectly assumed that they both had the same multiple professions [...] accurately identifying their professions.
(e) Next Trajectory	[...] Action: take pan 1 from stoveburner 2 [...] Obs: You put the pan 1 in countertop 1.	[...] return 'Yes' if check(S1) or check(S2) else 'No'	Think: [...] So the profession John Lanchester and Alan Dean Foster have in common is novelist. Action: "novelist"

Modular Formulation of Reflexion

Actor, denoted as M_a , which generates text and actions

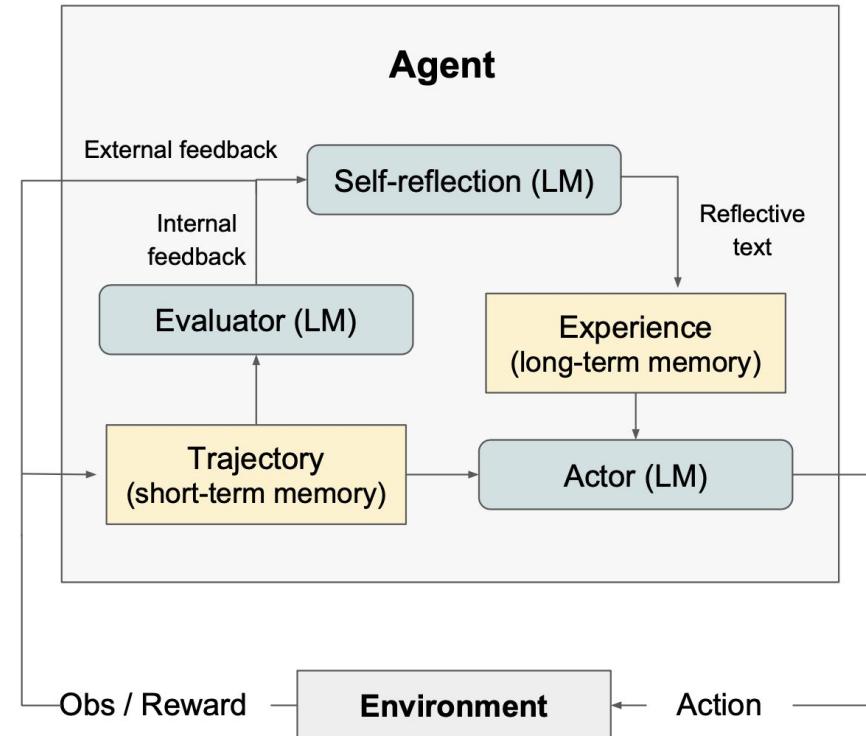
Evaluator model, represented by M_e , that scores the outputs produced by M_a

Self-Reflection model, denoted as M_{sr} , which generates verbal reinforcement cues to assist the Actor in self-improvement

Actor

Actor models could be Chain of Thought,
ReAct, etc.

We do **NOT** want to **finetune** the model!



Evaluator

Assessing the **quality** of the generated outputs produced by the Actor.

e.g. for **reasoning tasks**, the reward function can be based on **exact match (EM)** grading.

Using a different instantiation of an **LLM itself** as an Evaluator, generating rewards for **decision-making** and **programming** tasks.

Self-Reflection

It's just another name for “**Credit Assignment**”

Given a **sparse reward signal**, self-reflection produces a more **nuanced** feedback.

It is more **informative** than scalar rewards, is then stored in the agent’s **memory** (mem).

e.g. in a multi-step decision-making task, when receive a failure signal, infer that a specific action a_i led to subsequent incorrect actions a_{i+1} and a_{i+2} .

The agent can then **verbally state** that it should have taken a **different action**, a'_i , which **would have resulted in** a'_{i+1} and a'_{i+2} , and store this experience in its **memory**.

Algorithm 1 Reinforcement via self-reflection

Initialize Actor, Evaluator, Self-Reflection:
 M_a, M_e, M_{sr}

Initialize policy $\pi_\theta(a_i|s_i)$, $\theta = \{M_a, mem\}$

Generate initial trajectory using π_θ

Evaluate τ_0 using M_e

Generate initial self-reflection sr_0 using M_{sr}

Set $mem \leftarrow [sr_0]$

Set $t = 0$

while M_e not pass or $t < \text{max trials}$ **do**

- Generate $\tau_t = [a_0, o_0, \dots, a_i, o_i]$ using π_θ
- Evaluate τ_t using M_e
- Generate self-reflection sr_t using M_{sr}
- Append sr_t to mem
- Increment t

end while

return

Results

Two self-evaluation techniques:

- natural language classification using an LLM
- hand-written heuristic: if the agent executes the same action and receives the **same response for more than 3 cycles**, or if the **number of actions** taken in the current environment **exceeds 30** (inefficient planning), we self-reflect.

Results

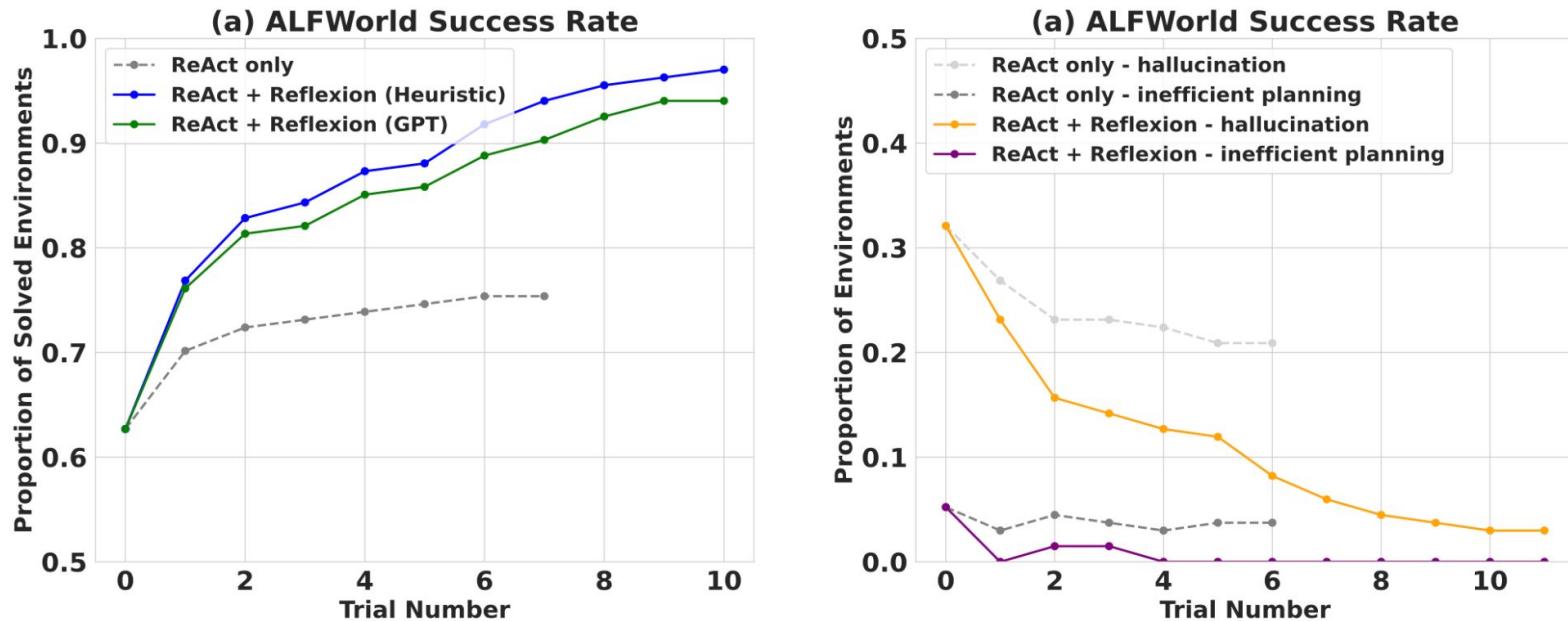


Figure 3: (a) AlfWorld performance across 134 tasks showing cumulative proportions of solved tasks using self-evaluation techniques of (Heuristic) and (GPT) for binary classification. (b) Classification of AlfWorld trajectories by reason of failure.

An Important Insight

A common error in baseline failed AlfWorld trajectories is when an agent thinks that it has possession of an item but does not actually have the item.

The agent proceeds to execute several actions in a long trajectory and is not able to backtrack its actions to find the mistake.

Reflexion eliminates almost all of these cases by using self-reflection to **distill long, failed trajectories into relevant experiences** that can be used as "self-hints" in the future.

In Code Generation

Benchmark + Language	Prev SOTA Pass@1	SOTA Pass@1	Reflexion Pass@1
HumanEval (PY)	65.8 (CodeT [5] + GPT-3.5)	80.1 (GPT-4)	91.0
HumanEval (RS)	–	60.0 (GPT-4)	68.0
MBPP (PY)	67.7 (CodeT [5] + Codex [6])	80.1 (GPT-4)	77.1
MBPP (RS)	–	70.9 (GPT-4)	75.4
Leetcode Hard (PY)	–	7.5 (GPT-4)	15.0

Table 1: Pass@1 accuracy for various model-strategy-language combinations. The base strategy is a single code generation sample. All instruction-based models follow zero-shot code generation.

Approach	Test Generation	Self-reflection	Pass@1 (Acc)
Base model	False	False	0.60
Test generation omission	False	True	0.52
Self-reflection omission	True	False	0.60
Reflexion	True	True	0.68

Table 3: Pass@1 accuracy for various compromised approaches on the Reflexion approach using GPT-4 as the base model on HumanEval Rust - 50 hardest problems

Issues with RAG

- Current RAG methods are unable to help LLMs perform tasks that require integrating new knowledge **across passage boundaries**.
 - each new passage is encoded in isolation.
- These require **knowledge integration across passages or documents**
 - scientific literature review,
 - legal case briefing,
 - and medical diagnosis.
- Standard **multi-hop question answering** (QA) also requires integrating information **between passages** in a retrieval corpus.

Long-term Memory

HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models

Bernal Jiménez Gutiérrez
The Ohio State University

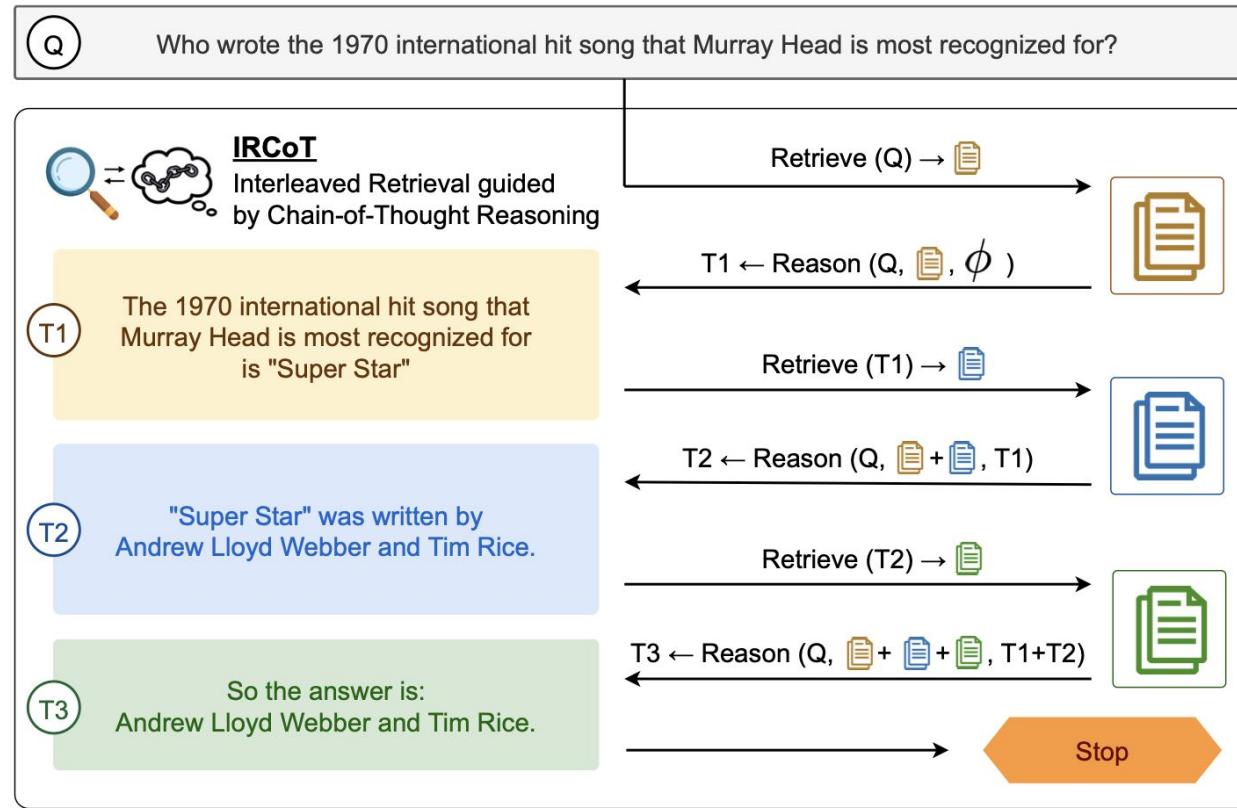
Yiheng Shu
The Ohio State University

Yu Gu
The Ohio State University

Michihiro Yasunaga
Stanford University

Yu Su
The Ohio State University

Previous Method (IRCoT)



Motivation

This is **challenging** for RAG to answer: “What Stanford professors do research on Alzheimer?”

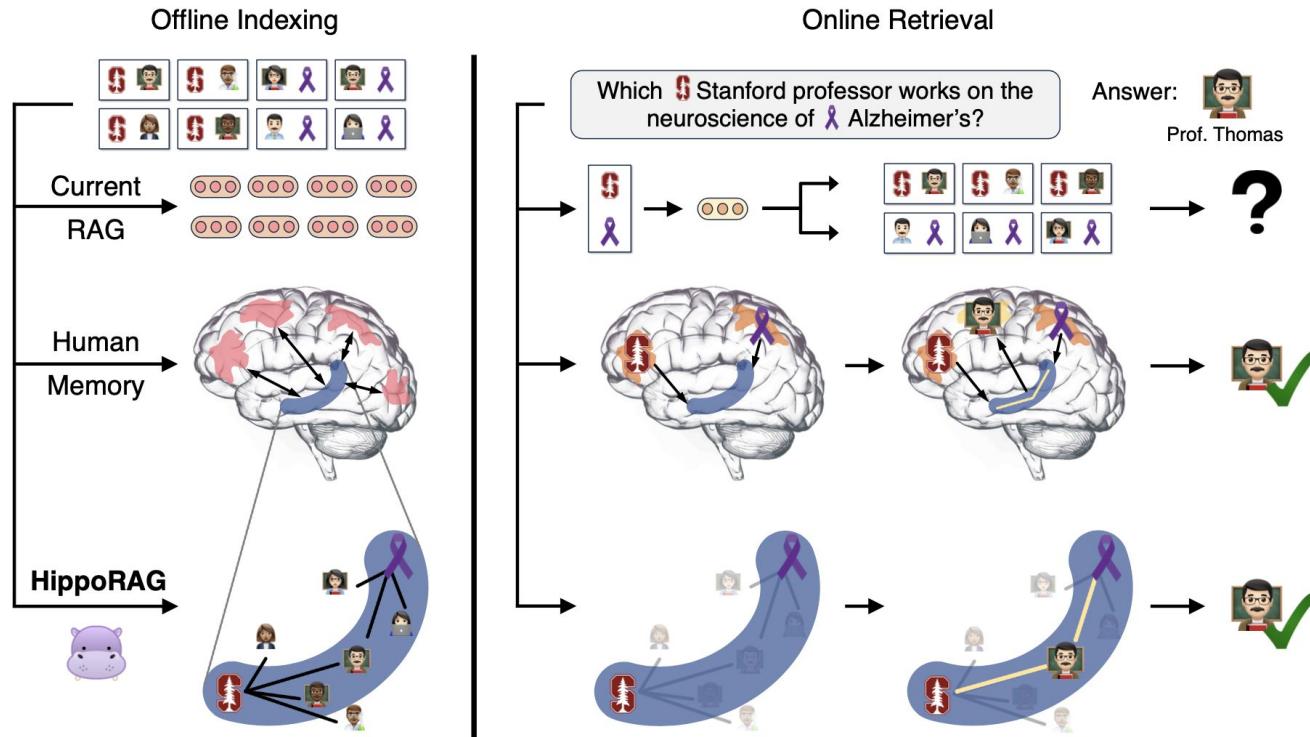
But not for **human**.

“our powerful context-based, continually updating memory relies on interactions between the

neocortex (storing actual memory representations), and

C-shaped hippocampus (holding the hippocampal index), a set of interconnected indices which point to memory units on the neocortex and stores associations between them.”

Hippocampal Memory Indexing Theory



Offline Indexing

- Passages P ; LLM L ; Retrieval Encoder M
- Use L to extract a set of noun phrase nodes N and relation edges E from each passage in P
- First extract a set of **named entities** from each passage
- Then add the named entities to the **OpenIE** prompt to **extract the final triples**.
- Use M to add the extra set of synonymy relations E' discussed above when the cosine similarity between two entity representations in N is above a threshold T

Passage NER (Indexing)

Instruction:

Your task is to extract named entities from the given paragraph.
Respond with a JSON list of entities.

One-Shot Demonstration:

Paragraph:

```

Radio City

Radio City is India's first private FM radio station and was started on 3 July 2001. It plays Hindi, English and regional songs. Radio City recently forayed into New Media in May 2008 with the launch of a music portal - PlanetRadiocity.com that offers music related news, videos, songs, and other music-related features.

```

```
{"named_entities": ["Radio City", "India", "3 July 2001", "Hindi", "English", "May 2008",  
"PlanetRadiocity.com"]}
```

Input:

Paragraph:

```

**PASSAGE TO INDEX**

```

Open Information Extraction

Instruction:

Your task is to construct an RDF (Resource Description Framework) graph from the given passages and named entity lists.

Respond with a JSON list of triples, with each triple representing a relationship in the RDF graph.

Pay attention to the following requirements:

- Each triple should contain at least one, but preferably two, of the named entities in the list for each passage.
- Clearly resolve pronouns to their specific names to maintain clarity.

Convert the paragraph into a JSON dict, it has a named entity list and a triple list.

One-Shot Demonstration:

Paragraph:

Radio City

Radio City is India's first private FM radio station and was started on 3 July 2001. It plays Hindi, English and regional songs. Radio City recently forayed into New Media in May 2008 with the launch of a music portal - PlanetRadiocity.com that offers music related news, videos, songs, and other music-related features.

```
{"named_entities": ["Radio City", "India", "3 July 2001", "Hindi", "English", "May 2008",  
"PlanetRadiocity.com"]}
```

{"triples":

[

- ["Radio City", "located in", "India"],
- ["Radio City", "is", "private FM radio station"],
- ["Radio City", "started on", "3 July 2001"],

Input:

Convert the paragraph into a JSON dict, it has a named entity list and a triple list.

Paragraph:

```

**PASSAGE TO INDEX**

```

{"named_entities": [NER LIST]}

Online Retrieval

- Identify all **query entities** using the LLM L.
- Use M to find entities in the KG **closest** to each query entity.

Query NER (Retrieval)

Instruction:

You're a very effective entity extraction system. Please extract all named entities that are important for solving the questions below. Place the named entities in JSON format.

One-Shot Demonstration:

Question: Which magazine was started first Arthur's Magazine or First for Women?

```
{"named_entities": ["First for Women", "Arthur's Magazine"]}
```

Input:

Question: QUERY TO INDEX

Personalized PageRank

Given source s , target t , and ‘teleport probability’ α

- Start a random walk from s .
- At each step, stop with probability α , else continue.

Then Personalized PageRank from s to t is given by:

$$\pi_s(t) = \mathbb{P} [\text{Walk from } s \text{ stops at } t]$$

- Equivalent to eigenvector/stationary distribution definitions
- FAST-PPR allows arbitrary starting set, e.g.
 - random $s \in V \implies$ Global PageRank
 - random $s \in S \implies$ personalize to S

Online Retrieval

- Run Personalized PageRank over **query nodes** to obtain **probability distribution n** over entities.
- Node specificity: $s_i = |P_i|^{-1}$, P_i be the **passages containing entity i** i. $n_i \leftarrow n_i s_i$
- These results in **higher probability** over nodes in joint neighborhood of query nodes.
- Let **C** be an $|N| \times |P|$ matrix, which contains the number of times each noun phrase in the KG appears in each original passage.
- Obtain passage score by $n \cdot C$.
- Let the context be **passages with top scores**.

Results

Table 2: **Single-step retrieval performance.** HippoRAG outperforms all baselines on MuSiQue and 2WikiMultiHopQA and achieves comparable performance on the less challenging HotpotQA dataset.

	MuSiQue		2Wiki		HotpotQA		Average	
	R@2	R@5	R@2	R@5	R@2	R@5	R@2	R@5
BM25 [69]	32.3	41.2	51.8	61.9	55.4	72.2	46.5	58.4
Contriever [35]	34.8	46.6	46.6	57.5	57.2	75.5	46.2	59.9
GTR [53]	37.4	49.1	60.2	67.9	59.4	73.3	52.3	63.4
ColBERTv2 [70]	37.9	49.2	59.2	68.2	64.7	79.3	53.9	65.6
RAPTOR [71]	35.7	45.3	46.3	53.8	58.1	71.2	46.7	56.8
RAPTOR (ColBERTv2)	36.9	46.5	57.3	64.7	63.1	75.6	52.4	62.3
Proposition [10]	37.6	49.3	56.4	63.1	58.7	71.1	50.9	61.2
Proposition (ColBERTv2)	37.8	50.1	55.9	64.9	63.9	78.1	52.5	64.4
HippoRAG (Contriever)	41.0	52.1	71.5	89.5	59.0	76.2	<u>57.2</u>	<u>72.6</u>
HippoRAG (ColBERTv2)	<u>40.9</u>	<u>51.9</u>	<u>70.7</u>	<u>89.1</u>	<u>60.5</u>	<u>77.7</u>	57.4	72.9

Results

Table 4: **QA performance.** HippoRAG’s QA improvements correlate with its retrieval improvements on single-step (rows 1-3) and multi-step retrieval (rows 4-5).

Retriever	MuSiQue		2Wiki		HotpotQA		Average	
	EM	F1	EM	F1	EM	F1	EM	F1
None	12.5	24.1	31.0	39.6	30.4	42.8	24.6	35.5
ColBERTv2	15.5	26.4	33.4	43.3	43.4	57.7	30.8	42.5
HippoRAG (ColBERTv2)	<u>19.2</u>	29.8	<u>46.6</u>	<u>59.5</u>	41.8	55.0	<u>35.9</u>	<u>48.1</u>
IRCoT (ColBERTv2)	19.1	<u>30.5</u>	35.4	45.1	<u>45.5</u>	<u>58.4</u>	33.3	44.7
IRCoT + HippoRAG (ColBERTv2)	21.9	33.3	47.7	62.7	45.7	59.2	38.4	51.7

NEXUS: A LIGHTWEIGHT AND SCALABLE MULTI-AGENT FRAMEWORK FOR COMPLEX TASKS AUTOMATION

**Humza Sami¹, Mubashir ul Islam¹, Samy Charas¹
Asav Gandhi¹, Pierre-Emmanuel Gaillardon^{1,2}, Valerio Tenace^{1*}**

¹PrimisAI, Los Gatos, CA, USA

²University of Utah, Salt Lake City, UT, USA

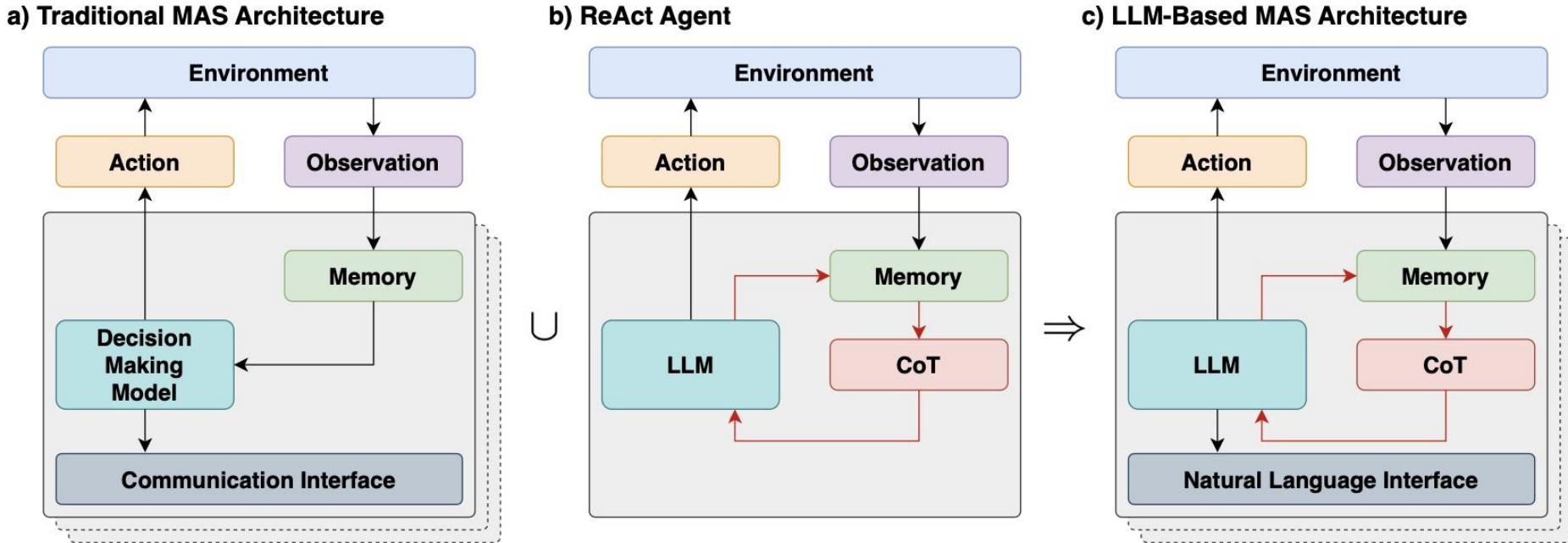
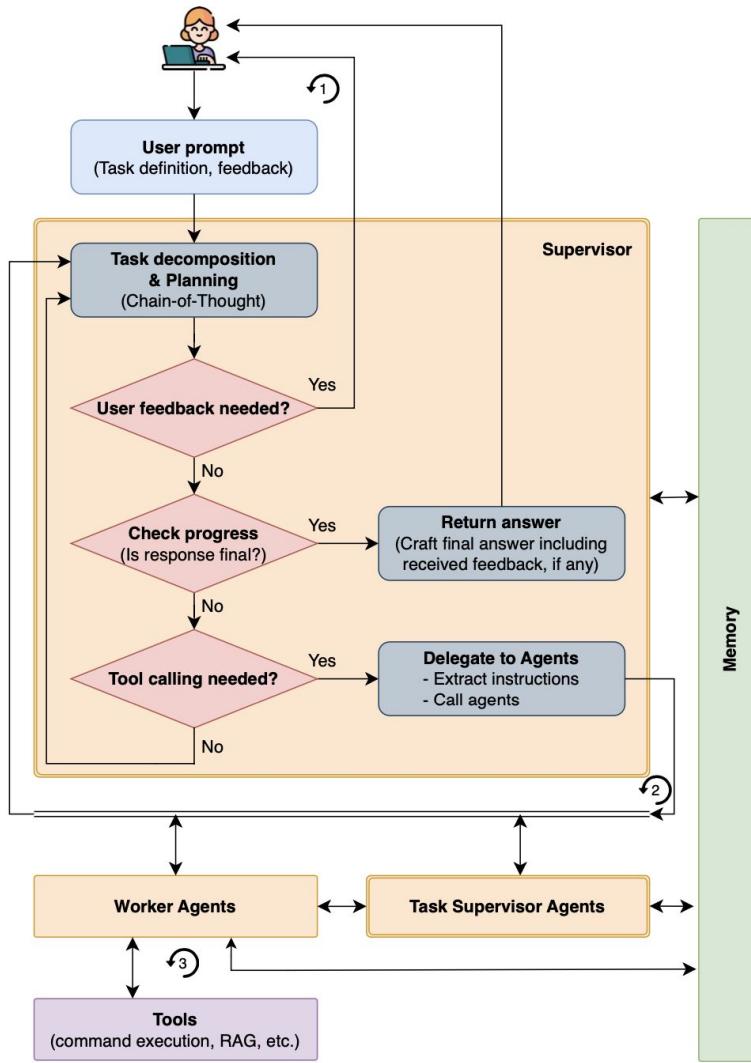


Figure 1: Evolution of Multi-Agent System Architectures: a) *Traditional MAS Architecture*, where agents interact with their environment through observations and actions; b) *ReAct Architecture*, an innovative agent design that incorporates advanced reasoning capabilities; and c) *LLM-Based MAS Architecture*, a cutting-edge approach leveraging LLMs for reasoning and decision-making.



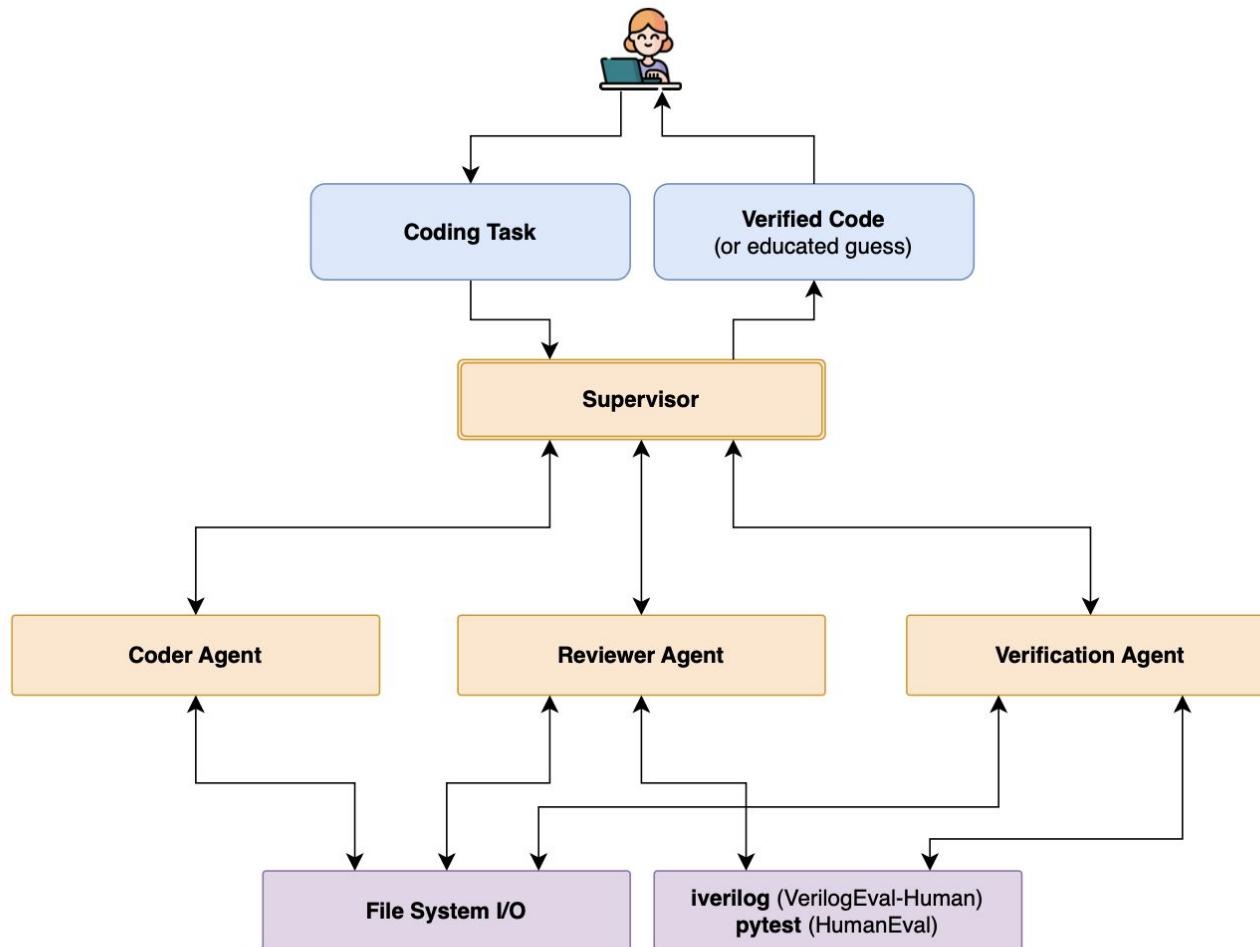


Figure 3: Unified Nexus-based MAS architecture for solving code-related tasks.

Results

Table 1 summarizes the effectiveness of the proposed self-verifying workflow. For both benchmark suites, we report the Claude 3.5 version used in each experiment (column *LLM Version*), along with the syntax pass rate (columns \mathcal{A}_s) and functional pass rate (\mathcal{A}_f) achieved by both the baseline Claude 3.5 and by Nexus employing the same underlying LLM.

Benchmark Suite	LLM Version	Baseline		Nexus (self-verifying)		
		\mathcal{A}_s	\mathcal{A}_f	\mathcal{A}_s	\mathcal{A}_f	$\Delta\mathcal{A}_f$
HumanEval	Claude 3.5 v1	99.39	87.80	100	96.95	$\uparrow 10.42\%$
HumanEval	Claude 3.5 v2	98.78	92.07	100	98.78	$\uparrow 7.28\%$
VerilogEval-Human	Claude 3.5 v2	90.38	67.30	100	85.90	$\uparrow 27.63\%$

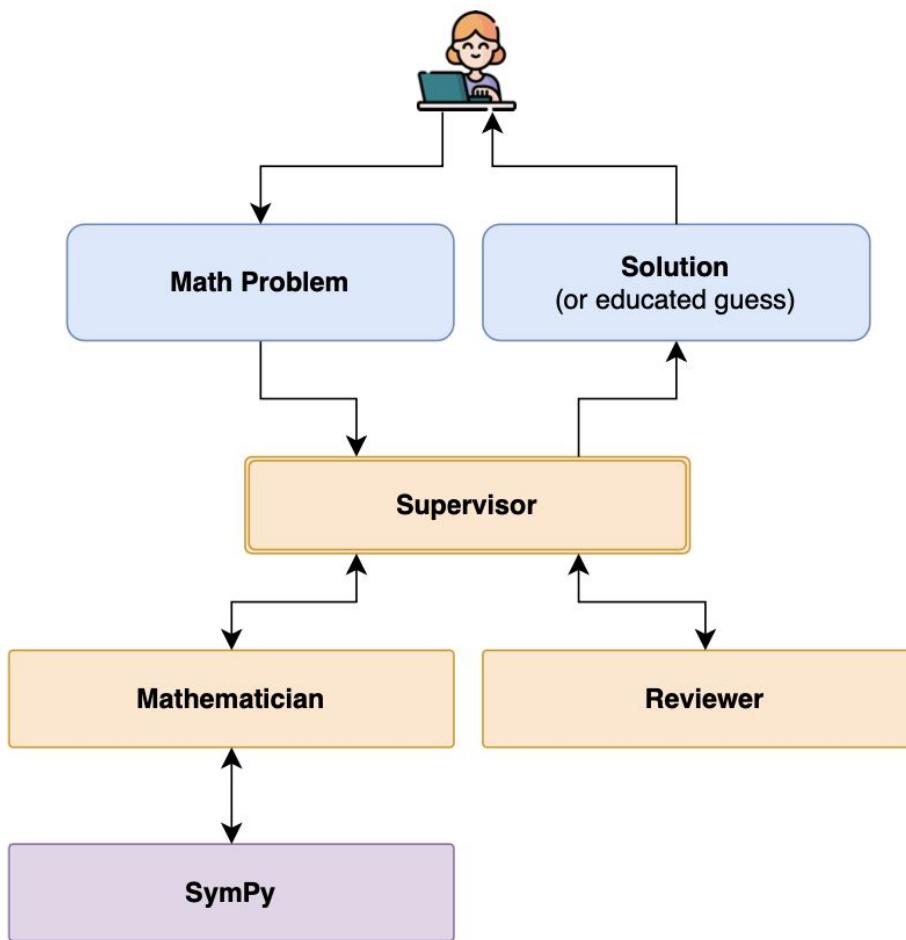


Figure 5: Proposed Nexus-based MAS architecture for solving problems from the MATH dataset.

Problem #25780. While staying in a 15-story hotel, Polya plays the following game. She enters an elevator on the 6th floor. She flips a fair coin five times to determine her next five stops. Each time she flips heads, she goes up one floor. Each time she flips tails, she goes down one floor. What is the probability that each of her next five stops is on the 7th floor or higher? Express your answer as a common fraction.

This problem challenges the solver to account for several key aspects: the starting position (6th floor), the probabilistic nature of movements determined by coin flips, the constraint of remaining on the 7th floor or higher for all five stops, and the need to compute the probability of all valid sequences. In essence, the complexity arises from the interplay between the random walks induced by coin flips and the geometric constraints imposed by the floor numbers, thus resulting in a particularly challenging task for typical LLMs.

In this scenario, the baseline LLM erroneously concluded that all five moves must be upward, resulting in a probability of $(\frac{1}{2})^5 = \frac{1}{32}$. The model's reasoning is summarized as follows:

CoT Excerpt—Baseline (Claude 3.5 v2). Assuming each heads (H) moves up one floor and each tails (T) moves down one floor, to ensure we're always at floor 7 or higher:

- First flip must be H
- Second flip must be H
- Third flip must be H
- Fourth flip must be H
- Fifth flip must be H

We need all heads for this to work. Any tails at any point would drop us below floor 7 at some step.

This demonstrates the limitations of LLMs in complex probability calculations without external aids. The baseline model failed to consider valid scenarios where some downward movements are possible while still maintaining the condition of staying on the 7th floor or higher. In contrast, the proposed Nexus-based workflow, while initially struggling, eventually produced the correct answer. What follows is the last review phase submitted by the *Reviewer* agent to the *Supervisor*:

CoT Excerpt—Nexus (Reviewer) After carefully reviewing this final solution, here is my detailed analysis:

1. Systematic Sequence Analysis:

- [OK] Correctly starts with H (required to reach 7th floor)
- [OK] Systematically examines all possible sequences starting with H
- [OK] Properly tracks floor numbers at each stop
- [OK] Good organization showing each sequence's path

2. Validity Check of Each Sequence:

- [OK] HHHHH: 7,8,9,10,11 (all stops ≥ 7)
- [OK] HHHHT: 7,8,9,10,9 (all stops ≥ 7)
- [OK] HHHTH: 7,8,9,8,9 (all stops ≥ 7)
- [OK] HHHTT: 7,8,9,8,7 (all stops ≥ 7)
- [OK] HHTHH: 7,8,7,8,9 (all stops ≥ 7)
- [OK] HHTHT: 7,8,7,8,7 (all stops ≥ 7)
- [OK] Correctly identifies invalid sequences (HHTTH, HHTTT, HTHXX, HTXXX)

3. Completeness Check:

- [OK] All possible sequences starting with H are accounted for
- [OK] Correctly eliminates sequences going below 7th floor
- [OK] Properly branches through all possibilities
- [OK] No valid sequences appear to be missed

4. Verification Process:

- [OK] Shows floor numbers at each stop
- [OK] Clear marking of valid ([OK]) and invalid ([X]) sequences
- [OK] Explains why invalid sequences are rejected
- [OK] Demonstrates that all 6 valid sequences stay ≥ 7 th floor

5. Probability Calculation:

- [OK] Correct denominator: $2^5 = 32$ (all possible 5-move sequences)
- [OK] Correct numerator: 6 (verified valid sequences)
- [OK] Correct reduction: $\frac{6}{32} = \frac{3}{16}$

Overall Assessment:

This solution is mathematically correct and complete. It:

- Uses a systematic approach
- Correctly identifies all valid sequences
- Properly verifies the ≥ 7 th floor requirement
- Accurately calculates the final probability
- Provides clear reasoning and documentation

The final answer of $\frac{3}{16}$ is correct and properly justified.