COMP0087

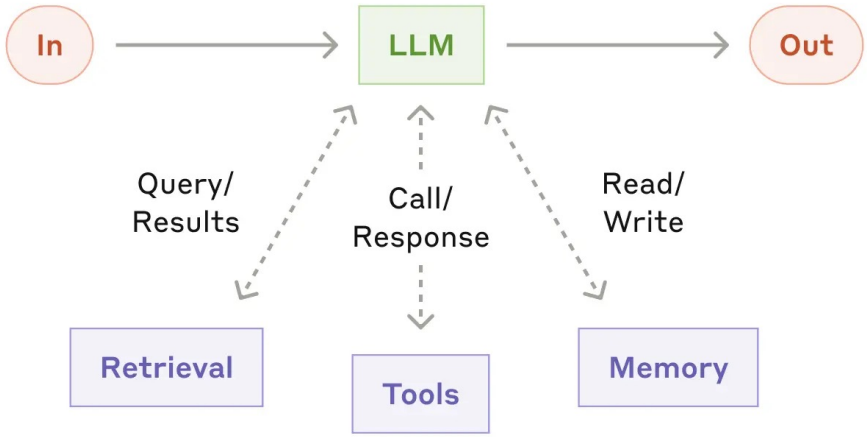
Statistical Natural Language Processing

Lecture 10 – LLM Augmentations

Overview

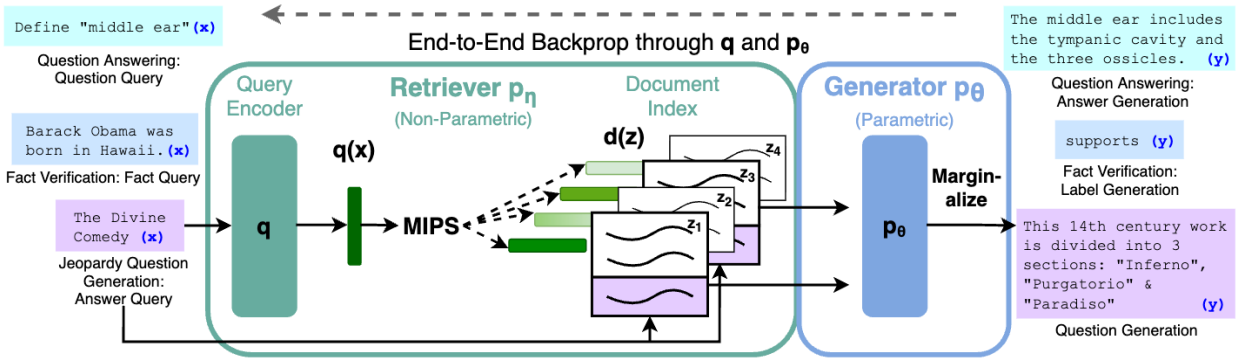
- Retrieval Augmented Generation (RAG)
- Tools
- Self-Refinements/Self-Correction
- Language Agents
  - Memory
  - Environment
  - Agentic Workflow
- Some interesting Trends

LLM Augmentations



RAG – Retrieval Augmented Generation

Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks



"We endow pre-trained, parametric-memory generation models with a non-parametric memory through a general-purpose fine-tuning approach which we refer to as retrieval-augmented generation (RAG)."

22 May 2020

See also references therein.

https://arxiv.org/pdf/2005.11401v4

Why RAG?

- Provides a pathway to expand knowledge recency
- Provides a pathway to incorporate domain-specific and/or sensitive data without training on them
- Could potentially reduce hallucination as outputs are grounded on existing facts/evidence
- Could attribute outputs to existing facts/evidence

Basic Components of RAG

Knowledge Base

Source of information

- Documents
- Databases
- Text corpus
- Chunked & indexed

Retrieval System

Finds relevant information from knowledge base

- Vector database
- Semantic search
- Document embeddings

Generator (LLM)

Creates final response

- Uses retrieved context
- Combines with query
- Produces grounded answer

Common approaches for encoding text into searchable vectors

1. Dense Embeddings (Most Common)

Semantic vectors from neural networks (384-1536 dims)
Models: OpenAI ada-002, Sentence-BERT, all-MiniLM

3. Hybrid (Dense + Sparse)

Combines semantic understanding + keyword matching
Best retrieval accuracy for production systems

2. Sparse Embeddings

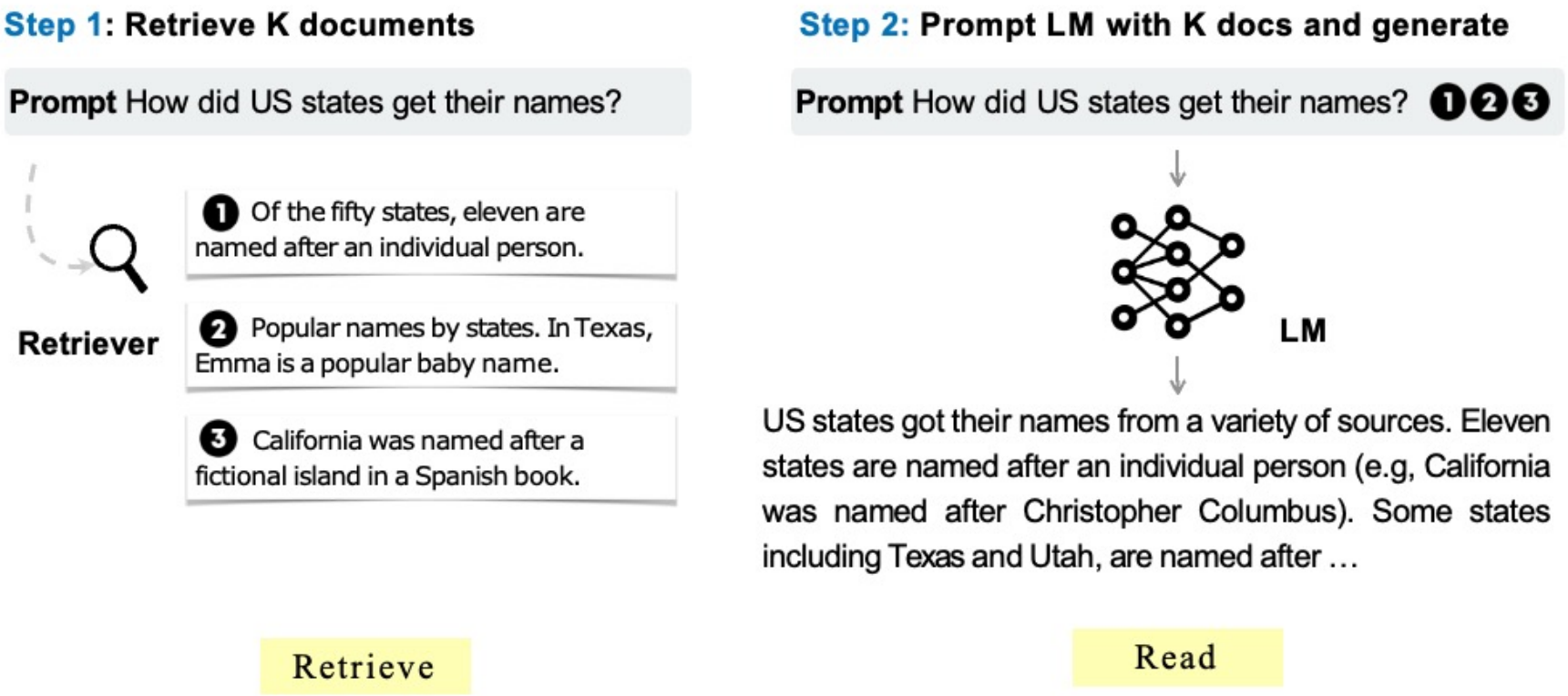
Keyword-based representations (TF-IDF, BM25)
Best for exact term matching in legal/technical docs

4. Multi-Vector & Contextual

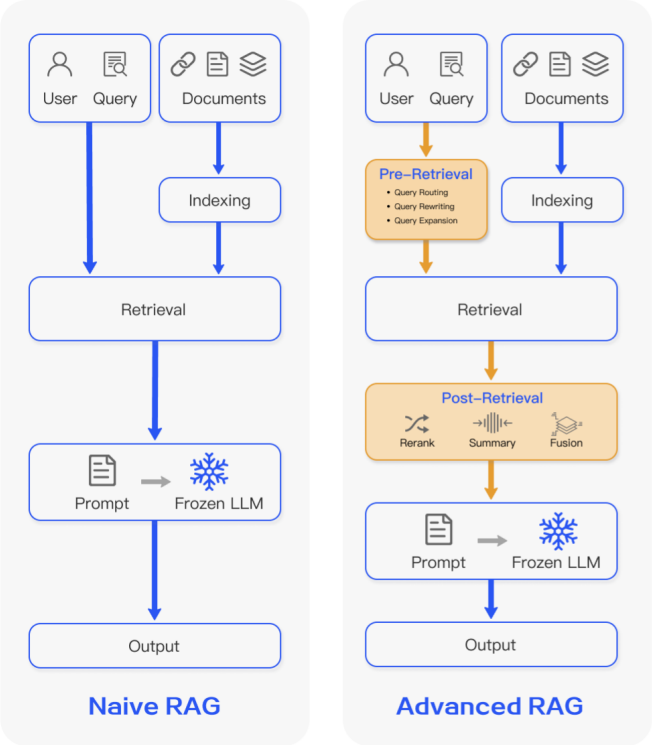
Multiple embeddings per doc (ColBERT, late interaction)
Handles polysemy and fine-grained semantic matching

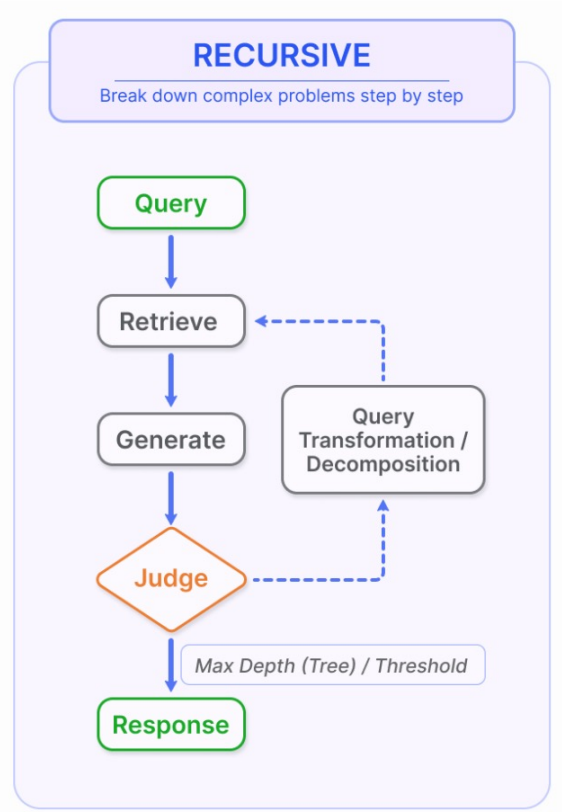
You can play with a simple notebook here.

Retrieval-Augmented Generation



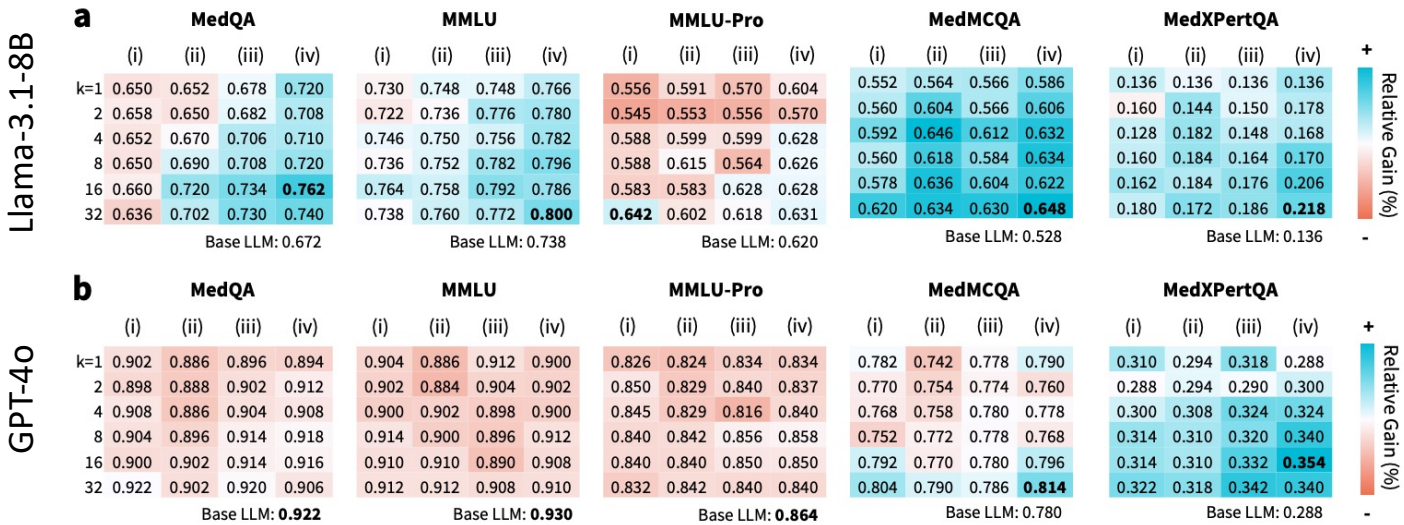
Versions of RAGs





[Retrieval-Augmented Generation for Large Language Models: A Survey - Gao et al., 2024]

Case study for medical domain

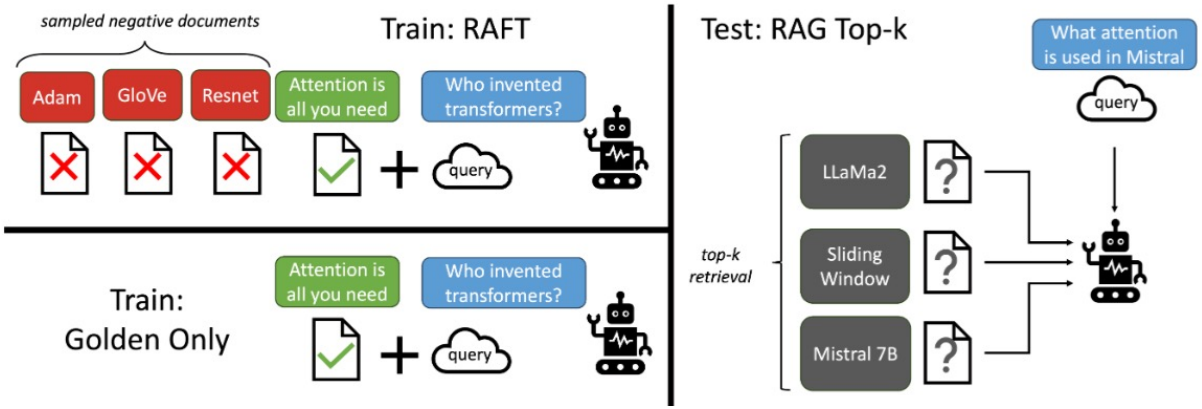


Cell colors indicate the magnitude of accuracy changes relative to the base LLM: sky-blue denotes gains, red indicates drops.

(i) standard RAG (retrieval-only),
(ii) retrieval + evidence filtering,
(iii) retrieval + query reformulation,
(iv) retrieval + both evidence filtering and query reformulation, evaluated at different top-k settings

https://arxiv.org/abs/2511.06738

RAFT: Adapting Language Model to Domain Specific RAG



RAFT: Retrieval Augmented Fine-Tuning (RAFT), presents a novel recipe to prepare fine-tuning data to tailor the models for domain-specific open-book setting, equivalent to in-domain RAG In RAFT, we prepare the training data such that each data point contains a question (Q), a set of documents (D_k), and a corresponding Chain-of-though style answer (A^*) generated from one of the document (D^*). We differentiate between two types of documents: 'golden' documents ($D*$) i.e. the documents from which the answer to the question can be deduced, and 'distractor' documents (D_i) that do not contain answer-

	PubMed	HotPot	HuggingFace
GPT-3.5 + RAG	71.60	**41.5**	29.08
LLaMA2-7B	56.5	0.54	0.22
LLaMA2-7B + RAG	58.8	0.03	26.43
DSF	59.7	6.38	61.06
DSF + RAG	71.6	4.41	42.59
RAFT (LLaMA2-7B)	**73.30**	35.28	**74.00**

$$\mathbf{P}\ \%\ \text{of data: } \mathbf{Q} + \mathbf{D}^* + \mathbf{D}_1 + \mathbf{D}_2 + \ldots + \mathbf{D}_k \rightarrow \mathbf{A}*$$

$$(1 - \mathbf{P})\ \%\ \text{of data: } \mathbf{Q} + \mathbf{D}_1 + \mathbf{D}_2 + \ldots + \mathbf{D}_k \rightarrow \mathbf{A}*$$

RAFT: Adapting Language Model to Domain Specific RAG (Zhang et al. 2024)

Modern RAG relies on:

- (1) Retrieval mechanism, which depends on:
  - How you build your database (indexing protocol)
  - What you index and how you index it

- (2) LLM's ability to utilise the retrieved information
  - LLM's Context size
  - LLM's capability to utilise the context

Context Size

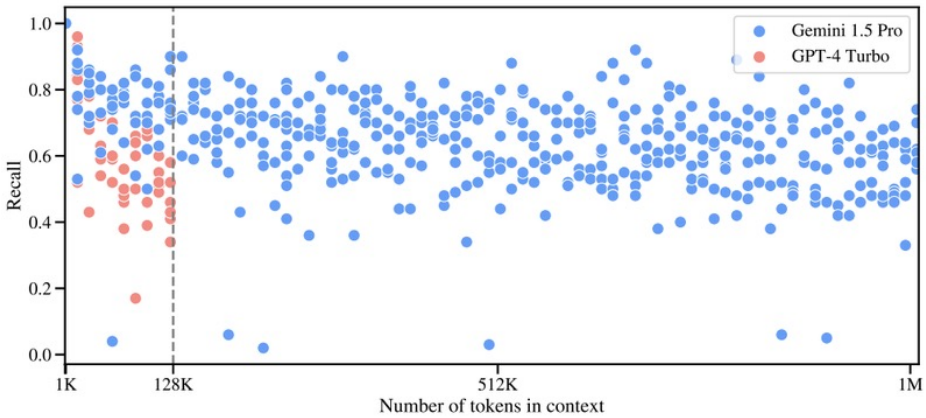
Outputs from retrieval could be a single or multiple hits.

They could each be a few words or several pages.

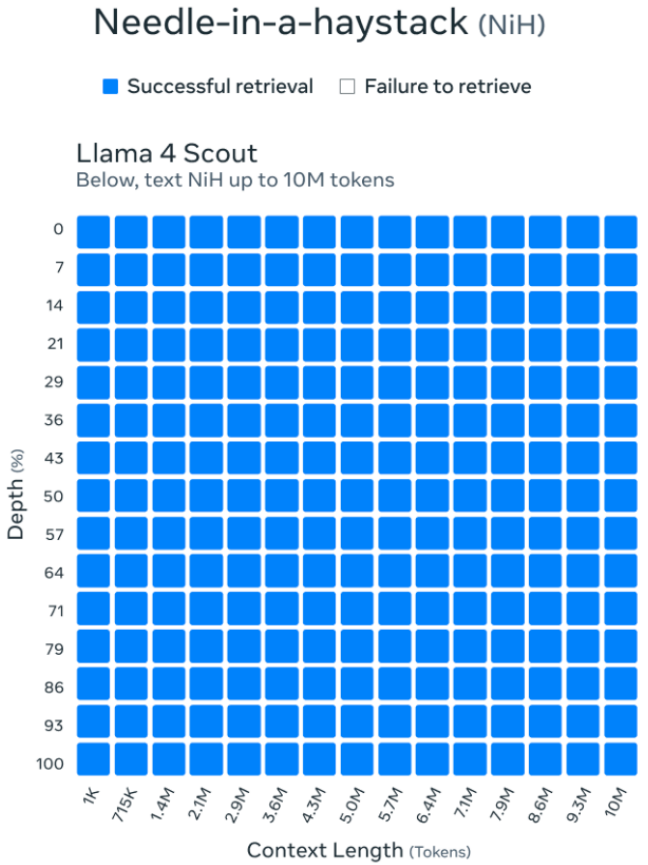
If we were to provide all hits to the generator (an LLM), we need to start worrying about context size limits of the LLM …

Needle in haystack

1. Place a random fact or statement (the "needle") in the middle of a long context window (the "haystack")
2. Ask the LLM to retrieve this statement
3. Iterate over various document depths (where the needle is placed) and context lengths to measure performance

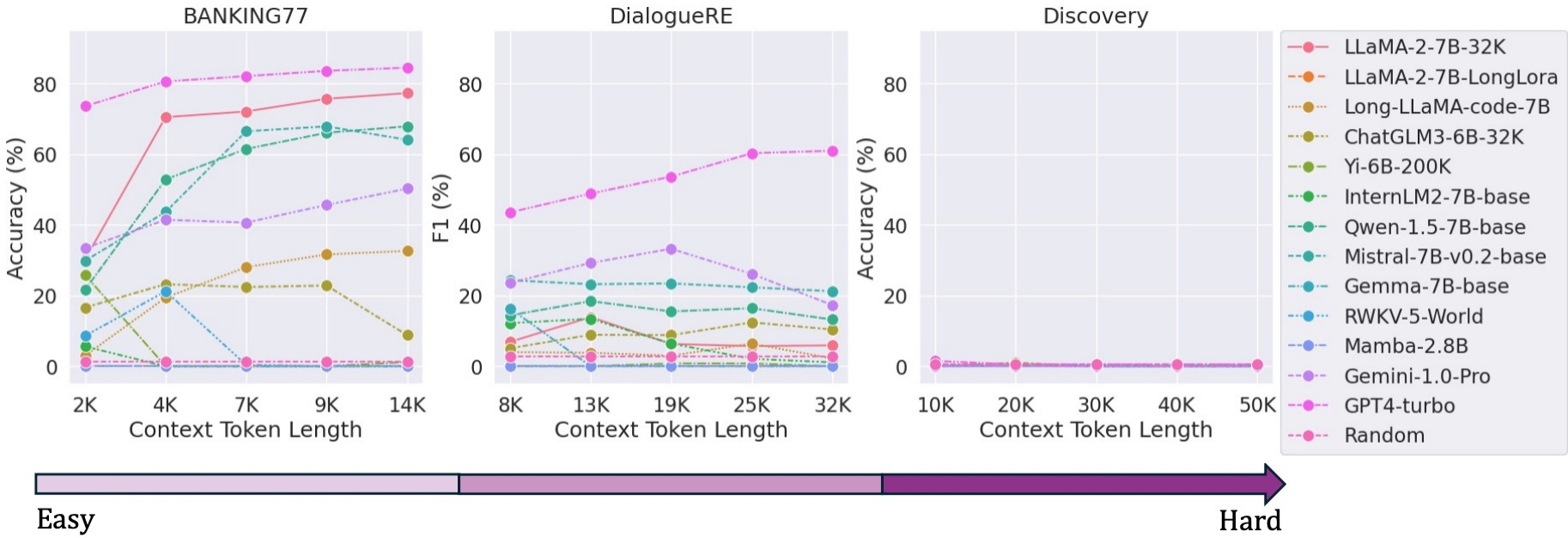


https://github.com/gkamradt/LLMTest_NeedleInAHaystack



https://ai.meta.com/blog/llama-4-multimodal-intelligence/

Retrieval is good, but LLM context size is limited, so we can't just throw long documents into prompt context. Long-context LLMs are out there (e.g., Gemini Pro 1.5 has 1-2M context window). But not good enough yet!



"we have discovered that while LLMs show promising performance on inputs up to 20K tokens, their ability to process and understand longer sequences significantly decreases"

Long-context LLMs Struggle with Long In-context Learning (Li, et al. 2024)

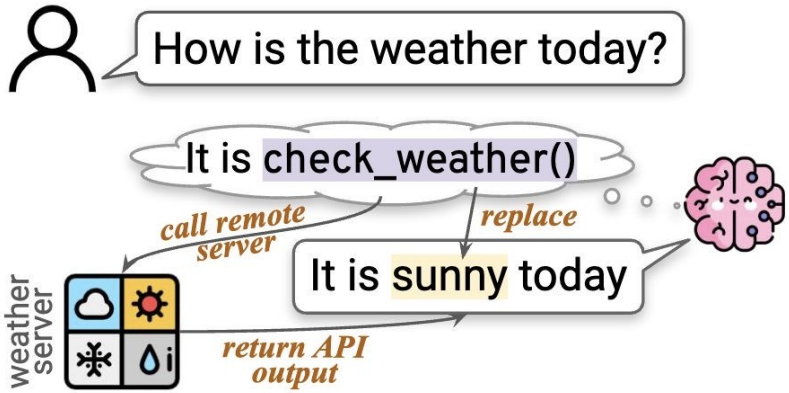
The Basic Tool Use Paradigm

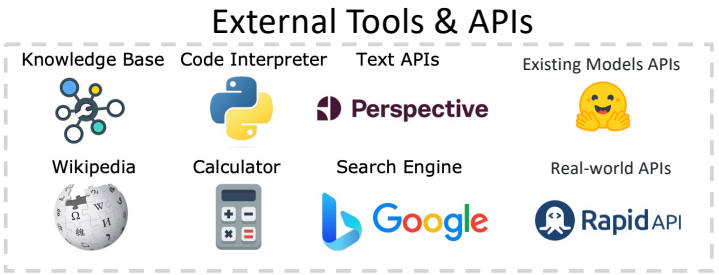
Tool Use: switching between

- text-generation mode
- tool-execution mode

Tool Learning:

- inference-time prompting
- learning by training





Why Tool-augmentation

- Leverage LLMs ability in writing structured outputs (i.e., code, JSON, SQL, etc)
- Enables new abilities provided by the specialised tools/APIs
- Improves the reliability of outputs as generated by tools/APIs (and not by a neural system)

Scenarios of LLM Tool Using

Category	Example Tools

Knowledge access	`sql_executor(query: str) -> answer: any`
	`search_engine(query: str) -> document: str`
	`retriever(query: str) -> document: str`

Computation activities	`calculator(formula: str) -> value: int | float`
	`python_interpreter(program: str) -> result: any`
	`worksheet.insert_row(row: list, index: int) -> None`

Interaction w/ the world	`get_weather(city_name: str) -> weather: str`
	`get_location(ip: str) -> location: str`
	`calendar.fetch_events(date: str) -> events: list`
	`email.verify(address: str) -> result: bool`

Non-textual modalities	`cat_image.delete(image_id: str) -> None`
	`spotify.play_music(name: str) -> None`
	`visual_qa(query: str, image: Image) -> answer: str`

Special-skilled LMs	`QA(question: str) -> answer: str` `translation(text: str, language: str) -> text: str`

HuggingGPT – ChatGPT to Plan and Call other APIs

HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face

Yongliang Shen[1]*, Kaitao Song[2]*, Xu Tan[2], Dongsheng Li[2], Weiming Lu[1], Yueting Zhuang[1]
Zhejiang University[1], Microsoft Research Asia[2]
{syl, luwm, yzhuang}@zju.edu.cn, {kaitaosong, xuta, dongsli}@microsoft.com

April 2023
https://arxiv.org/pdf/2303.17580.pdf

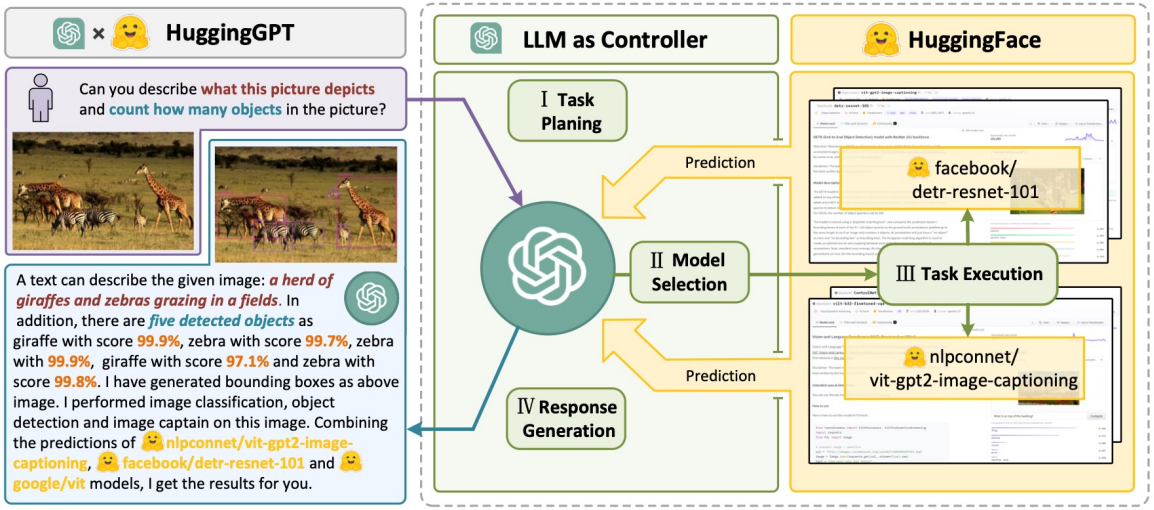
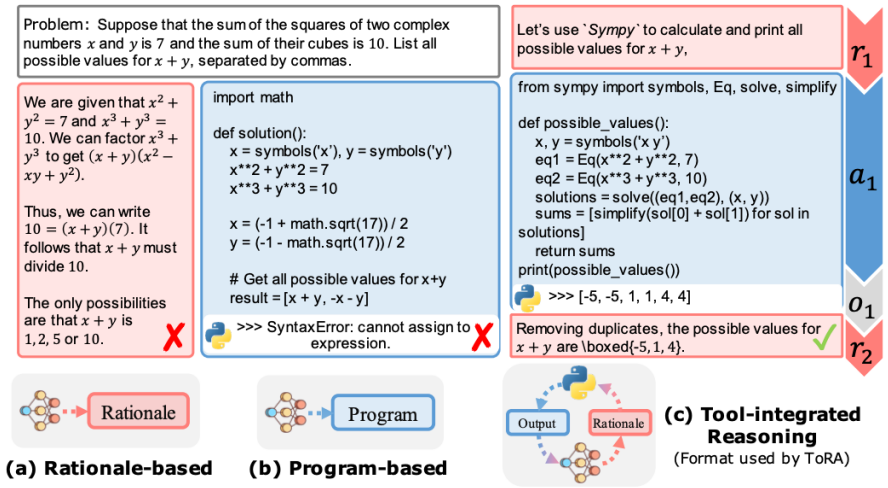


Figure 1: *Language serves as an interface for LLMs (e.g., ChatGPT) to connect numerous AI models (e.g., those in Hugging Face) for solving complicated AI tasks.* In this concept, an LLM acts as a controller, managing and organizing the cooperation of expert models. The LLM first plans a list of tasks based on the user request and then assigns expert models to each task. After the experts execute the tasks, the LLM collects the results and responds to the user.

Similar idea that created a massive wave:

Auto-GPT: An Autonomous GPT-4 Experiment
https://github.com/Significant-Gravitas/Auto-GPT

Tool-augmented Mathematical Reasoning



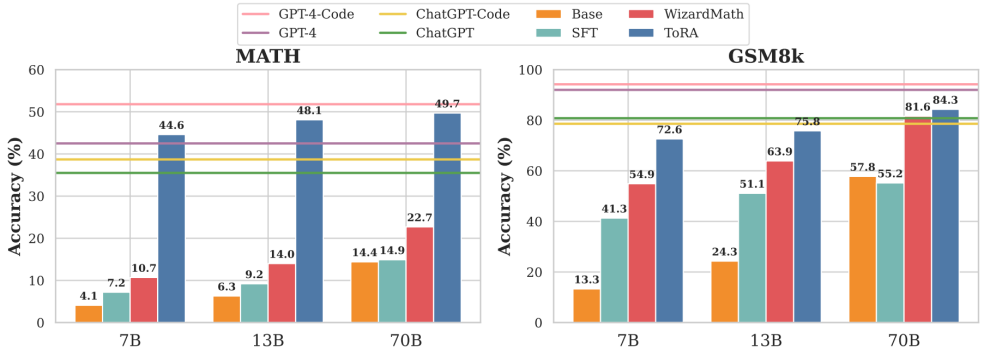
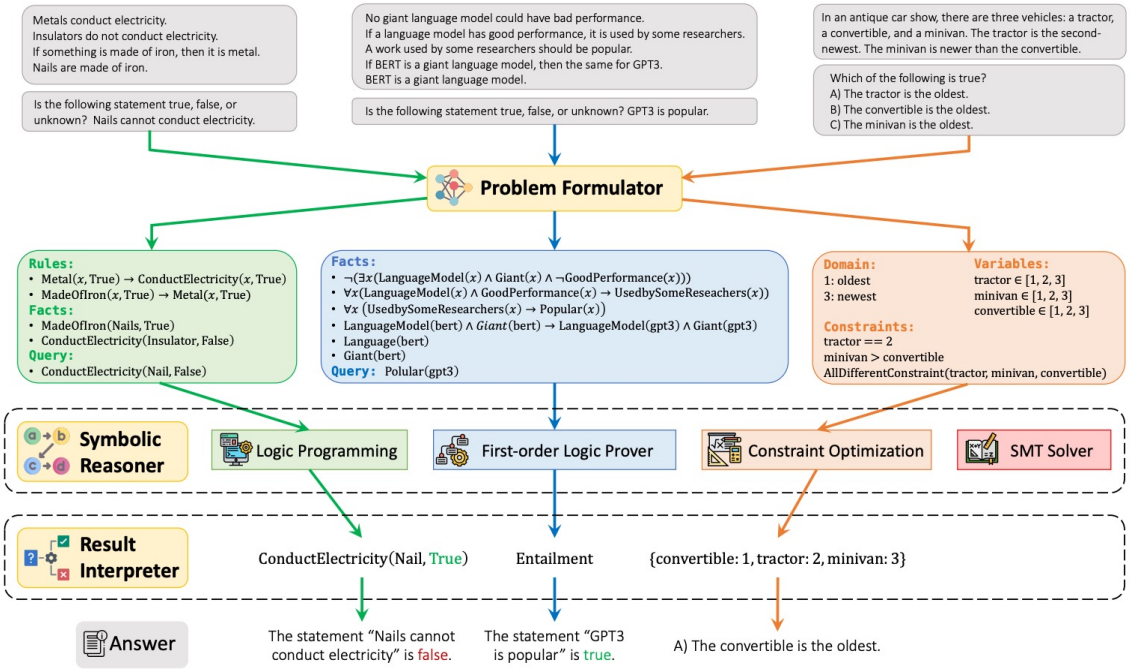


Figure 1: Comparing ToRA with baselines on LLaMA-2 base models from 7B to 70B. ToRA models exhibit remarkable improvements over previous state-of-the-art approaches across all scales. In particular, ToRA-70B notably outperforms GPT-4's CoT result on MATH and attains comparable results to GPT-4 solving problems with code.

[ToRA: A Tool-Integrated Reasoning Agent for Mathematical Problem Solving (Guo, et al. 2024)]

Tool-augmented Logical Reasoning

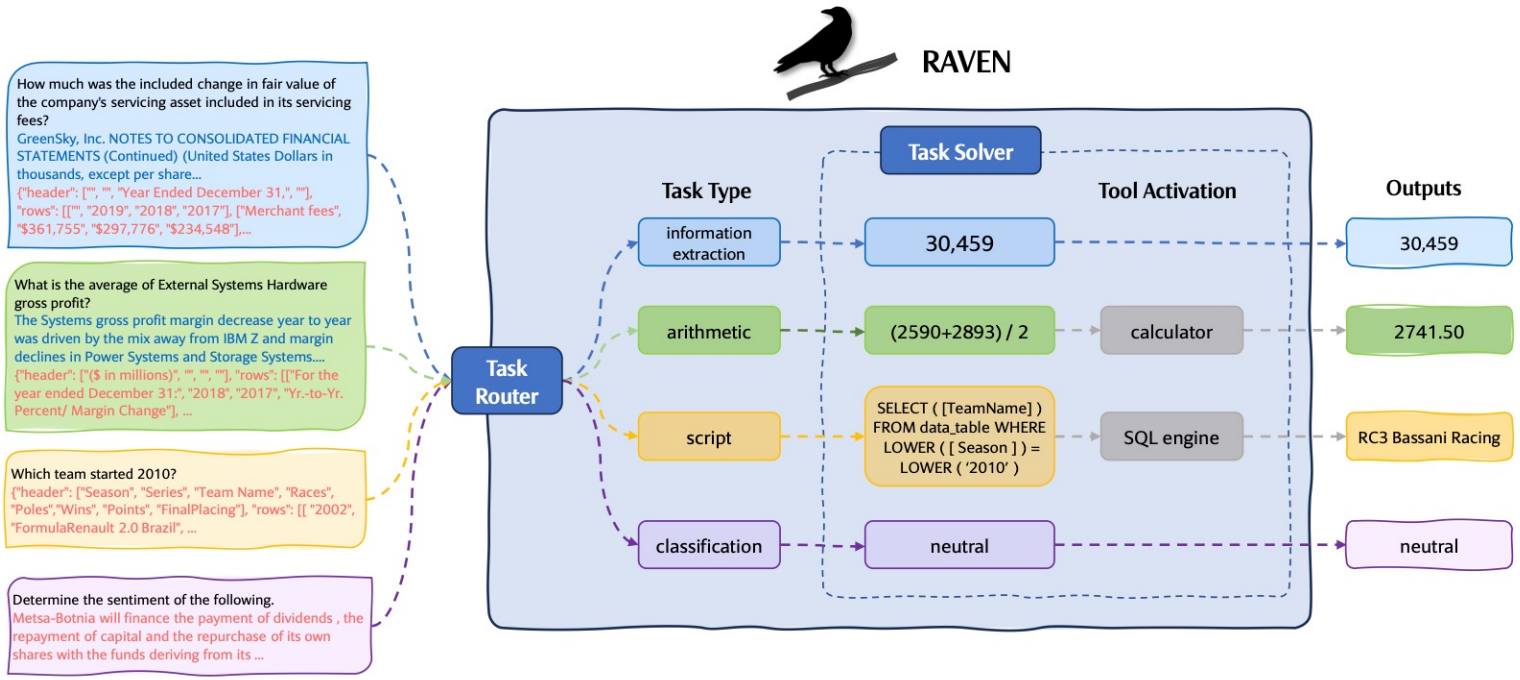


Dataset	ChatGPT (gpt-3.5-turbo)			GPT-3.5 (text-davinci-003)			GPT-4 (gpt-4)		
	Standard	CoT	Logic-LM	Standard	CoT	Logic-LM	Standard	CoT	Logic-LM
PrOntoQA	47.40	67.80	61.00	51.80	83.00	85.00	77.40	98.79	83.20
ProofWriter	35.50	49.17	58.33	36.16	48.33	71.45	52.67	68.11	79.66
FOLIO	45.09	57.35	62.74	54.60	57.84	61.27	69.11	70.58	78.92
LogicalDeduction	40.00	42.33	65.67	41.33	48.33	62.00	71.33	75.25	87.63
AR-LSAT	20.34	17.31	26.41	22.51	22.51	25.54	33.33	35.06	43.04

Table 2: Accuracy of standard promoting (Standard), chain-of-thought promoting (CoT), and our method (LOGIC-LM, without self-refinement) on five reasoning datasets. The best results within each base LLM are highlighted.

[Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning (Pan, et al. 2023)]

Tool-augmented Tabular Data Analysis for Finance



[Equipping Language Models with Tool Use Capability for Tabular Data Analysis in Finance (Theuma, et al 2024)]

Example of Data

[Equipping Language Models with Tool Use Capability for Tabular Data Analysis in Finance (Theuma, et al 2024)]

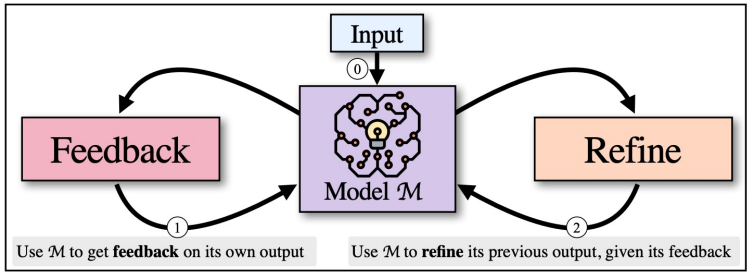
What was the change in the basic net earnings per share between 2017 and 2019?

Input:
(5) Earnings Per Share Basic earnings per share is computed by dividing Net earnings attributable to Black Knight by the weighted-average number of shares of common stock outstanding during the period. For the periods presented, potentially dilutive securities include unvested restricted stock awards and the shares of BKFS Class B common stock prior to the Distribution. For the year ended December 31, 2017, the numerator in the diluted net earnings per share calculation is adjusted to reflect our income tax expense at an expected effective tax rate assuming the conversion of the shares of BKFS Class B common stock into shares of BKFS Class A common stock on a one-for-one basis prior to the Distribution. The effective tax rate for the year ended December 31, 2017 was (16.7)%, including the effect of the benefit related to the revaluation of our net deferred income tax liability and certain other discrete items recorded during 2017. For the year ended December 31, 2017, the denominator includes approximately 63.1 million shares of BKFS Class B common stock outstanding prior to the Distribution. The denominator also includes the dilutive effect of approximately 0.9 million, 0.6 million and 0.6 million shares of unvested restricted shares of common stock for the years ended December 31, 2019, 2018 and 2017, respectively. The shares of BKFS Class B common stock did not share in the earnings or losses of Black Knight and were, therefore, not participating securities. Accordingly, basic and diluted net earnings per share of BKFS Class B common stock have not been presented. The computation of basic and diluted earnings per share is as follows (in millions, except per share amounts):

Data:
{"header": ["", "", "Year ended December 31,", ""], "rows": [["", "2019", "2018", "2017"], ["Basic:", "", "", ""], ["Net earnings attributable to Black Knight", "$108.8", "$168.5", "$182.3"], ["Shares used for basic net earnings per share:", "", "", ""], ["Weighted average shares of common stock outstanding", "147.7", "147.6", "88.7"], ["Basic net earnings per share", "$0.74", "$1.14", "$2.06"], ["Diluted:", "", "", ""], ["Earnings before income taxes and equity in losses of unconsolidated affiliates", "", "", "$192.4"], ["Income tax benefit excluding the effect of noncontrolling interests", "", "", "(32.2)"], ["Net earnings", "", "", "$224.6"], ["Net earnings attributable to Black Knight", "$108.8", "$168.5", ""], ["Shares used for diluted net earnings per share:", "", "", ""], ["Weighted average shares of common stock outstanding", "147.7", "147.6", "88.7"], ["Dilutive effect of unvested restricted shares of common", "", "", ""], ["stock", "0.9", "0.6", "0.6"], ["Weighted average shares of BKFS Class B common stock outstanding", "", "", "63.1"], ["Weighted average shares of common stock, diluted", "148.6", "148.2", "152.4"], ["Diluted net earnings per share", "$0.73", "$1.14", "$1.47"]]}

Equation:
0.74-2.06

Self-Reflection



Uses the same underlying LLM to generate feedback and refine its outputs

Self-refinement or self-correction approaches in LLMs enable models to:

- Learn from their interactions and correct their own errors
- Reduce the need for periodic retraining,
- Make LLMs more responsive to new information or changing environments,

Ultimately, these approaches could lead to more autonomous systems capable of continuous learning and improvement.

Self-Reflection

Apply a three-step prompting strategy for self-Reflection:

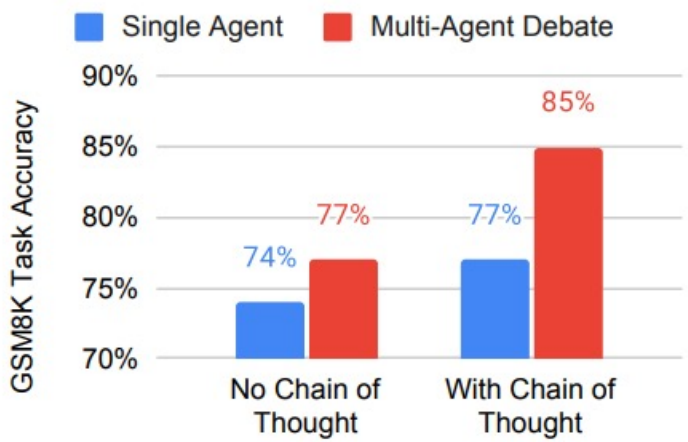
1) prompt the model to perform an initial generation;

2) prompt the model to review its previous generation and produce feedback;
   e.g., "Assume that this answer could be either correct or incorrect. Review the answer carefully and report any serious problems you find."

3) prompt the model to answer the original question again with the feedback.

Self-Refine: Iterative Refinement with Self-Feedback (Madaan, et al. 2023)

LLM Debate



Debate Length	Prompt
Short	*" These are the solutions to the problem from other agents: [other answers] Based off the opinion of other agents, can you give an updated response . . ."*
Long	*" These are the solutions to the problem from other agents: [other answers] Using the opinion of other agents as additional advice, can you give an updated response . . ."*

Figure 3: **Prompts to induce long and short form debate.** Responses of other agents to questions are are inserted in the middle of the prompt (indicated with *[other answers]*)

[Improving Factuality and Reasoning in Language Models through Multiagent Debate (Du, et al. 2024)]

User Input: A treasure hunter found a buried treasure chest filled with gems. There were 175 diamonds, 35 fewer rubies than diamonds, and twice the number of emeralds than the rubies. How many of the gems were there in the chest?

Round 1

Agent 1: Let x be the number of rubies. Then the number of diamonds is 175. And the number of emeralds is 2(x-35), since there are twice as many emeralds as rubies and there are 35 fewer rubies than diamonds.

To find the total number of gems, we add the number of diamonds, rubies, and emeralds: Total gems = 175 + x + 2(x-35) = 3x + 105.

We don't know the exact value of x, but we do know that it represents the number of rubies, which is less than 175 (since there are 35 fewer rubies than diamonds). Let's try some values of x: If x = 50, then the total gems = 3(50) + 105 = 255. If x = 100, then the total gems = 3(100) + 105 = 405.

Since we know there were fewer rubies than diamonds, the answer must be closer to 255 than to 405. After some trial and error, we can find that x = 40 gives us: Total gems = 3(40) + 105 = 225. ✗

Agent 2: If there were 175 diamonds, then the number of rubies is 35 fewer, or 175 - 35 = 140 rubies.

And since there were twice as many emeralds as rubies, we can represent the number of emeralds as 2 * 140 = 280 emeralds.

To find the total number of gems in the chest, we can add up the number of diamonds, rubies, and emeralds: 175 + 140 + 280 = 595. Therefore, there were 595 gems in the chest. ✓

User Input: Using the solutions from other agents as additional information, can you give an updated response....
(Copy Agent 2's reply) (Copy Agent 1's reply)

Round 2

Agent 1: Given the information provided in other agents' solutions, we have two answers: One agent found that the number of rubies is 40 and the total number of gems is 225. Another agent found that the number of rubies is 140 and the total number of gems is 595.

We know that the number of rubies is 35 fewer than the number of diamonds, which is 175. So the number of rubies should be 175 - 35 = 140, as the second agent found. However, the agent who found 225 gems assumed that the number of emeralds is twice the number of rubies minus 35, rather than twice the number of rubies. If we use this assumption, we get: Number of rubies = 140. Number of emeralds = 2 * 140 - 35 = 245. Total number of gems = 175 + 140 + 245 = 560. Our final answer is 560. ✗

Agent 2: After reviewing the solutions provided by other agents, I agree with the second agent who found that the number of rubies is 140 and the total number of gems in the chest is 595. Therefore, my answer to the math problem is 595. ✓

User Input: Using the solutions from other agents as additional information, can you give an updated response....
(Copy Agent 2's reply) (Copy Agent 1's reply)

Round 3

Agent 1: After reviewing the solutions provided by other agents, I agree with the second agent who found that the number of rubies is 140 and the total number of gems in the chest is 595. Therefore, my answer to the math problem is 595. ✓

Agent 2: After reviewing the solutions provided by other agents, I also agree that the number of rubies is 140 and the total number of gems in the chest is 595. Therefore, my answer to the math problem is also 595. ✓

Figure 2: **Illustration of Debate.** Illustration of the debate procedure.

LLM Debate

Level	Prompt
0	Both sides must reach a full consensus on every point of the debate. Every statement must be agreed upon by both sides.
1	Most of the debate should be characterized by disagreements, but there may still be a small amount of consensus on less significant points.
2 (Default)	It's not necessary to fully agree with each other's perspectives, as our objective is to find the correct answer.
3	Both sides must disagree with each other on every point of the debate. There should be no consensus whatsoever.

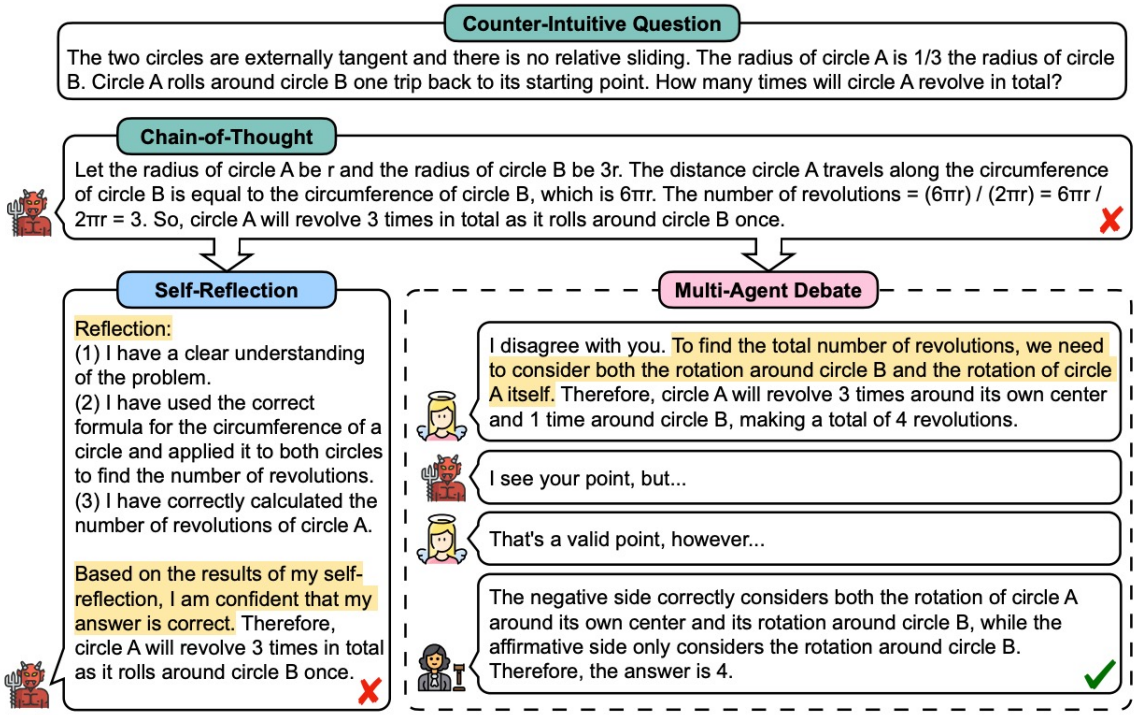


Figure 2: Framework of Multi-Agent Debate. Here we designate the devil ( ) as the affirmative side while the angel ( ) as the negative side. We want the angel to correct the devil's mistakes.

[Encouraging Divergent Thinking in Large Language Models through Multi-Agent Debate (Liang, et al. 2024)]

Can LLMs really self-correct?

" In the context of reasoning, … LLMs struggle to self-correct their responses without external feedback, and at times, their performance even degrades after self-correction."

Table 7: Results of multi-agent debate and self-consistency.

	# responses	GSM8K
Standard Prompting	1	76.7
Self-Consistency	3	82.5
Multi-Agent Debate (round 1)	6	83.2
Self-Consistency	6	85.3
Multi-Agent Debate (round 2)	9	83.0
Self-Consistency	9	**88.2**

Large Language Models Cannot Self-Correct Reasoning Yet (Huang, et al. 2024)

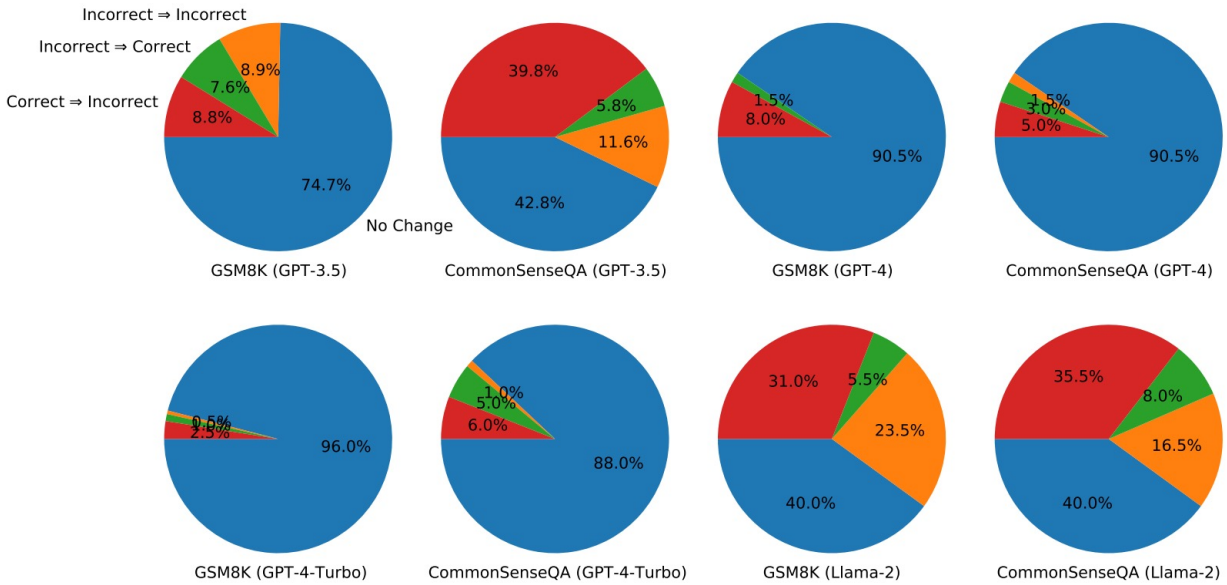


Figure 1: Analysis of the changes in answers after two rounds of self-correction. *No Change*: The answer remains unchanged; *Correct* ⇒ *Incorrect*: A correct answer is changed to an incorrect one; *Incorrect* ⇒ *Correct*: An incorrect answer is revised to a correct one; *Incorrect* ⇒ *Incorrect*: An incorrect answer is altered but remains incorrect.

Table 1: Summary of issues in previous LLM self-correction evaluation.

Method	Issue
RCI (Kim et al., 2023); Reflexion (Shinn et al., 2023)	Use of oracle labels (Section 3)
Multi-Agent Debate (Du et al., 2023)	Unfair comparison to self-consistency (Section 4)
Self-Refine (Madaan et al., 2023)	Sub-optimal prompt design (Section 5)

Simulating If-then

		GSM8K	SVAMP	HotpotQA	Sports	LLC	Average
GPT-3.5-0613	Standard Prompt	74.9	82.2	51.0	75.6	68.0	70.3
	+ Critical Prompt (Huang et al., 2023)	74.1	80.0	47.0	53.6	76.0	66.1
	+ IoE Prompt (Ours)	77.1	81.9	**55.0**	**77.1**	74.0	73.0
	+ IoE Prompt + Decision (Ours)	**78.5**	**83.3**	53.0	76.5	**77.3**	**73.7**
GPT-3.5-1106	Standard Prompt	80.1	82.9	61.0	74.1	41.3	67.9
	+ Critical Prompt (Huang et al., 2023)	77.3	81.5	54.0	68.4	40.7	64.4
	+ IoE Prompt (Ours)	80.9	83.2	62.0	**75.7**	38.7	68.1
	+ IoE Prompt + Decision (Ours)	**82.3**	**84.2**	**63.0**	74.7	**44.7**	**69.8**
GPT-4	Standard Prompt	92.5	92.8	68.0	80.7	91.3	85.1
	+ Critical Prompt (Huang et al., 2023)	88.4	89.5	62.0	82.9	89.9	82.5
	+ IoE Prompt (Ours)	93.4	**93.2**	**70.0**	83.1	93.3	86.6
	+ IoE Prompt + Decision (Ours)	**93.6**	93.1	**70.0**	**83.3**	**94.7**	**86.9**
Mistral-Medium	Standard Prompt	84.8	85.7	67.0	75.6	60.7	74.8
	+ Critical Prompt (Huang et al., 2023)	62.5	74.5	65.0	51.0	35.4	57.7
	+ IoE Prompt (Ours)	85.4	85.7	**68.0**	75.6	**61.3**	75.2
	+ IoE Prompt + Decision (Ours)	**85.6**	**85.8**	**68.0**	**75.9**	**61.3**	**75.3**

Table 5: **The accuracy comparisons between our IoE-based Prompt and the baseline Critical Prompt**. The results (%) are evaluated on 5 different benchmarks by 4 large models. IoE Prompt + Decision denotes further using the decision refinement stage. Our IoE Prompt achieves consistent improvement on all settings over standard prompt and the Critical Prompt baseline.

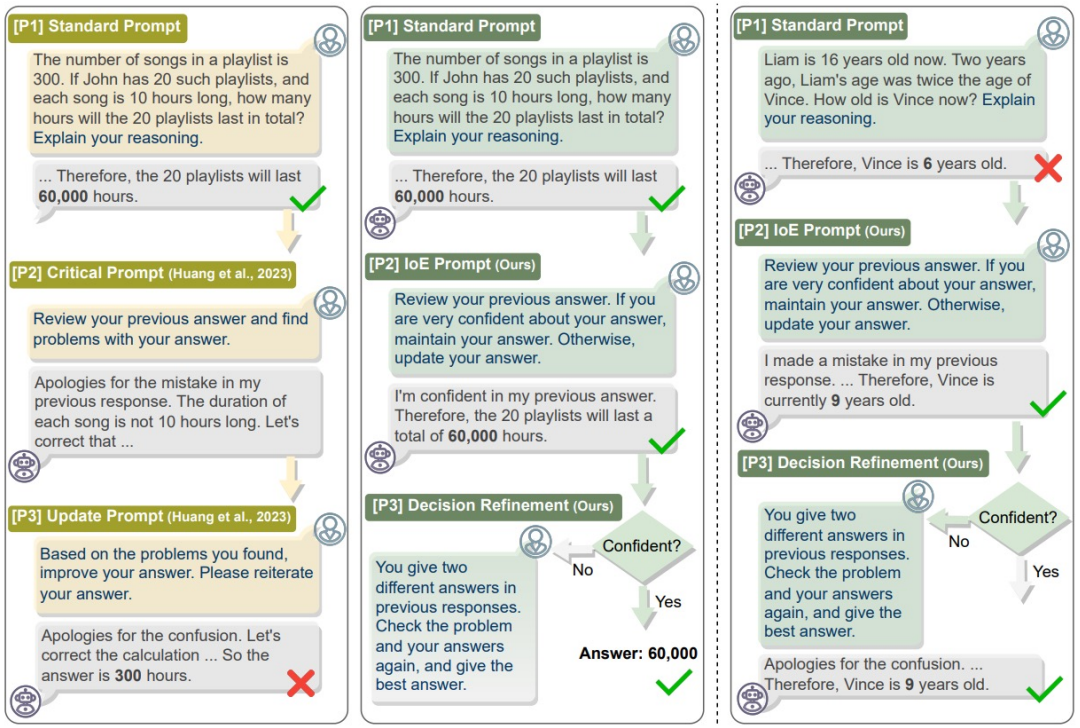
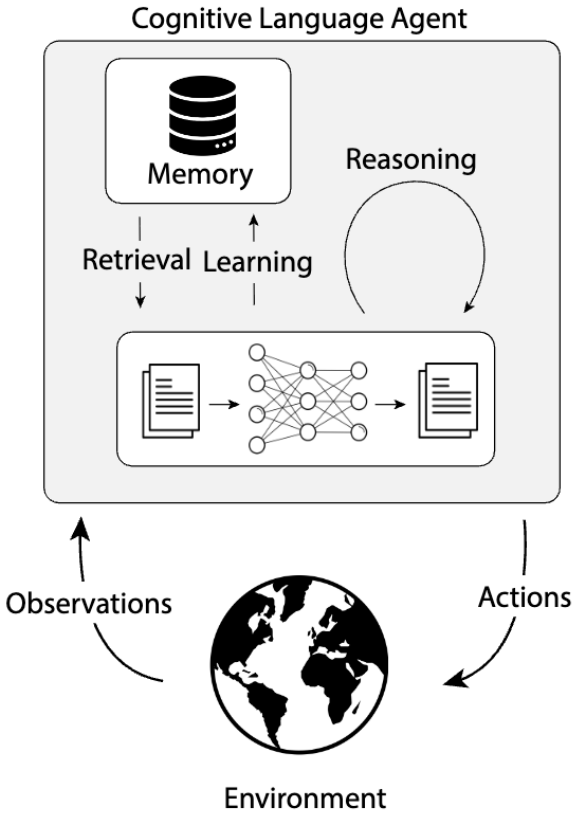


Figure 1: **Comparisons between our IoE-based Prompt and Critical Prompt**. **Left:** Critical Prompt (Huang et al., 2023) as baseline. **Middle/Right:** the proposed IoE-based Prompt. Regarding our prompts, when the answers of [P1] standard question and [P2] IoE prompt match, the final answer will be directly output, as shown in the **middle**. Otherwise, the decision prompt for final decision-making will execute, as the example shown in the **right**. All examples are generated from GSM8K (Cobbe et al., 2021) and evaluated by `gpt-3.5-turbo-1106` model.

Confidence Matters: Revisiting Intrinsic Self-Correction Capabilities of Large Language Models (Li, et al. 2024)

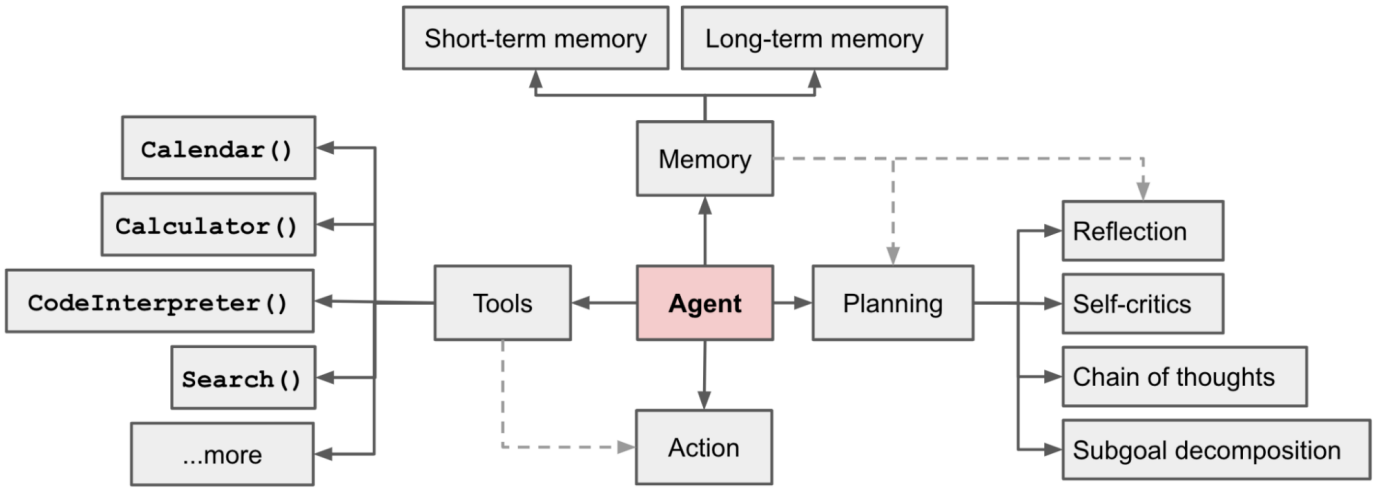
Language Agents

Language agents are an emerging class of AI systems that use LLMs to interact with the world. They apply the latest advances in LLMs to the existing field of agent design. LLMs possess limited knowledge and reasoning capabilities. Language agents mitigate these issues by connecting LLMs to internal memory and environments (tools, APIs, physical environment, human, etc), grounding them to existing knowledge or external observations. Language agents leverage commonsense priors present in LLMs to adapt to novel tasks, reducing the dependence on human annotation or trial-and-error learning.



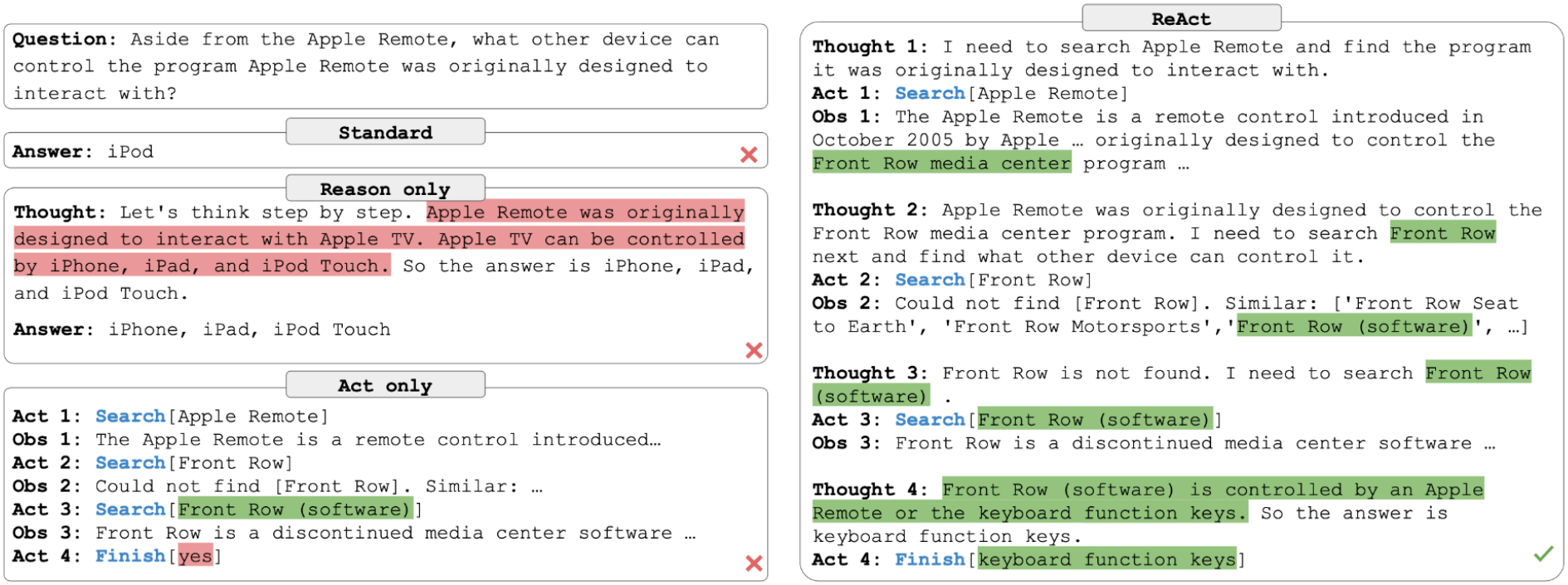
Cognitive Architectures for Language Agents https://arxiv.org/pdf/2309.02427.pdf

Language Agents



https://lilianweng.github.io/posts/2023-06-23-agent/

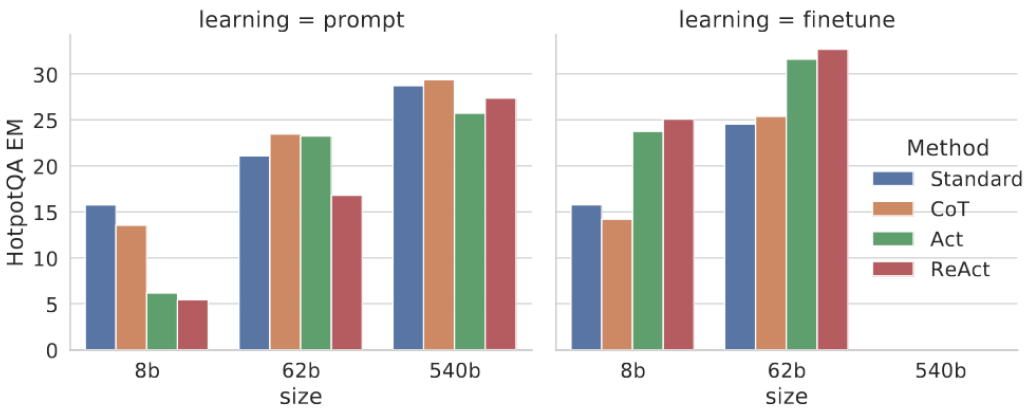
ReAct



ReAct: Synergizing Reasoning and Acting in Language Models - arXiv:2210.03629

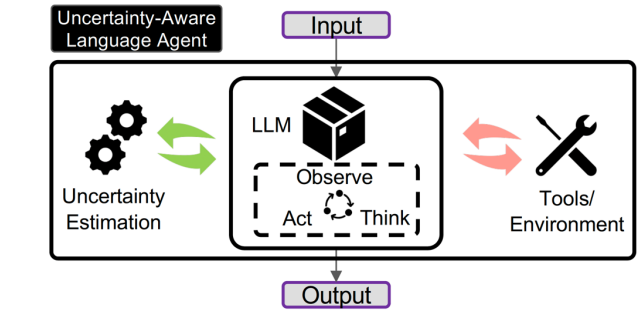
ReAct

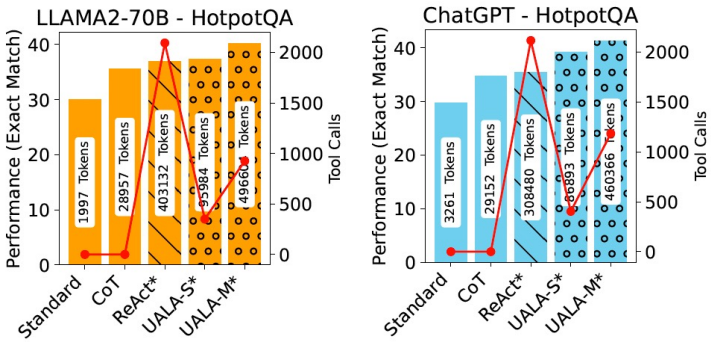
Prompt Method[a]	HotpotQA (EM)	Fever (Acc)
Standard	28.7	57.1
CoT (Wei et al., 2022)	29.4	56.3
CoT-SC (Wang et al., 2022a)	33.4	60.4
Act	25.7	58.9
ReAct	27.4	60.9
CoT-SC → ReAct	34.2	**64.6**
ReAct → CoT-SC	**35.1**	62.0
Supervised SoTA[b]	67.5	89.5



ReAct: Synergizing Reasoning and Acting in Language Models - arXiv:2210.03629

UALA





Question: Who is older Danny Green or James Worthy?

Thought 1: Let's think step by step. Danny Green was born in 1987. James Worthy was born in 1961. 1961 (James Worthy) < 1987 (Danny Green), so James Worthy is older than Danny Green. So the Answer is James Worthy.
Action 1: MeasureUncertainty [James Worthy].
Observation 1: Answer's uncertainty is 1.09 which falls within the acceptable threshold of 1.79.

Thought 2: Based on the uncertainty, answer is kept.
Action 2: Keep Answer.
Observation 2: []

Answer: James Worthy

(Example a)

Question: Paul Ziert recruited what two time gold medalist in gymnastics?

Thought 1: Let's think step by step. Paul Ziert is a coach for the U.S. national gymnastics team. The two-time gold medalist in gymnastics is Gabby Douglas. So Paul Ziert recruited Gabby Douglas. So the Answer is Gabby Douglas.
Action 1: MeasureUncertainty [Gabby Douglas].
Observation 1: Answer's uncertainty is 3.67 which falls outside the acceptable threshold of 1.79.

Thought 2: Based on the uncertainty, I need to use an external tool to solve the question.
Action 2: Activate Tool.
Observation 2: []

Thought 3: I need to search Paul Ziert ...
Action 3: Search [Paul Ziert]
Observation 3: ... He recruited Bart Conner to the school ...

Thought 4: ... I need to find who Bart Conner is.
Action 4: Search [Bart Conner]
Observation 4: ... Conner won two gold medals. ...

Thought 5: ... two time gold medalist in gymnastics that Paul Ziert recruited is Bart Conner.
Action 5: Finish [Bart Conner]
Observation 5: [Episode Finished]

Thought 6: Let me check the uncertainty of the returned answer.
Action 6: MeasureUncertainty [Bart Conner]
Observation 6: Answer's uncertainty is 1.10 which falls within the acceptable threshold of 1.79.

Thought 7: Based on the uncertainty, answer is kept.
Action 7: Keep Answer.
Observation 7: []

Answer: Bart Conner

(Example b)

Question: What schools of thought are drawn from in the academic discipline of Hafiz Rashid Khan?

Thought 1: Let's think step by step. Hafiz Rashid Khan is a scholar of Islamic studies. Islamic studies draws from various schools of thought, including theology, philosophy, and jurisprudence. Therefore, the schools of thought ... are theology, philosophy, and jurisprudence.
Action 1: MeasureUncertainty [theology, philosophy, jurisprudence].
Observation 1: Answer's uncertainty is 1.94 which falls outside the acceptable uncertainty threshold of 1.79.

Thought 2: Based on the uncertainty, I need to use an external tool to solve the question.
Action 2: Activate Tool.
Observation 2: []

Thought 3: I need to search Hafiz Rashid Khan, then find the schools of thought ...
Action 3: Search [Hafiz Rashid Khan]
Observation 3: ... is a Bangladeshi postcolonialist poet, author, editor, ...

Thought 4: ... His academic discipline is postcolonialist. ... find the schools of thought drawn from it.
Action 4: Search [postcolonialism schools of thought]
Observation 4: ... Could not find postcolonialism schools of thought. ...

Thought 5: I need to instead search postcolonial theory.
Action 5: Search [postcolonial theory]
Observation 5: Postcolonialism (also post-colonial theory) is the critical academic ...

Thought 6: The schools of thought drawn from postcolonial theory are critical theory, epistemology, ethics, and political science.
Action 6: Finish [critical theory, epistemology, ethics, political science]
Observation 6: [Episode Finished]

Thought 7: Let me check the uncertainty of the returned answer.
Action 7: MeasureUncertainty [critical theory, epistemology, ethics, political science]
Observation 7: Answer's uncertainty is 2.40 which falls outside the acceptable threshold of 1.79.

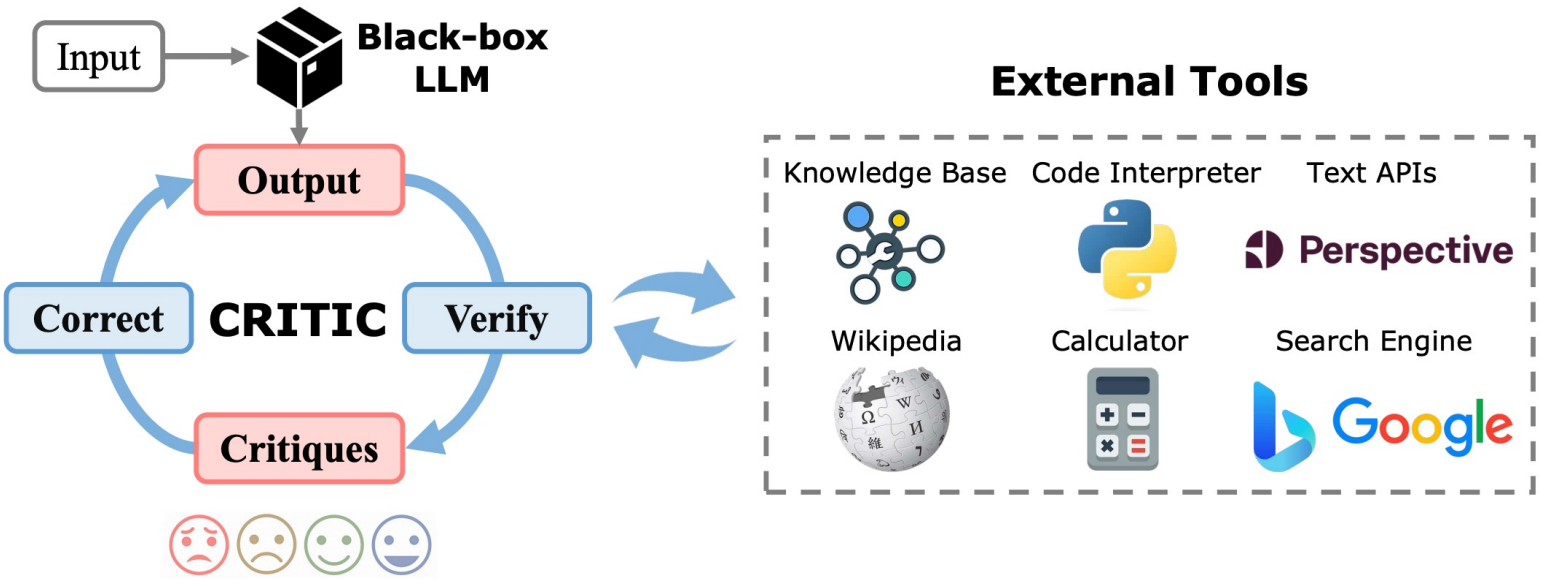
Thought 8: Based on the uncertainty, I need to ask a human for help.
Action 8: Ask Human.
Observation 8: postmodern schools of thought

Answer: postmodern schools of thought

(Example c)

[Towards Uncertainty-Aware Language Agent - Han et al., 2024]

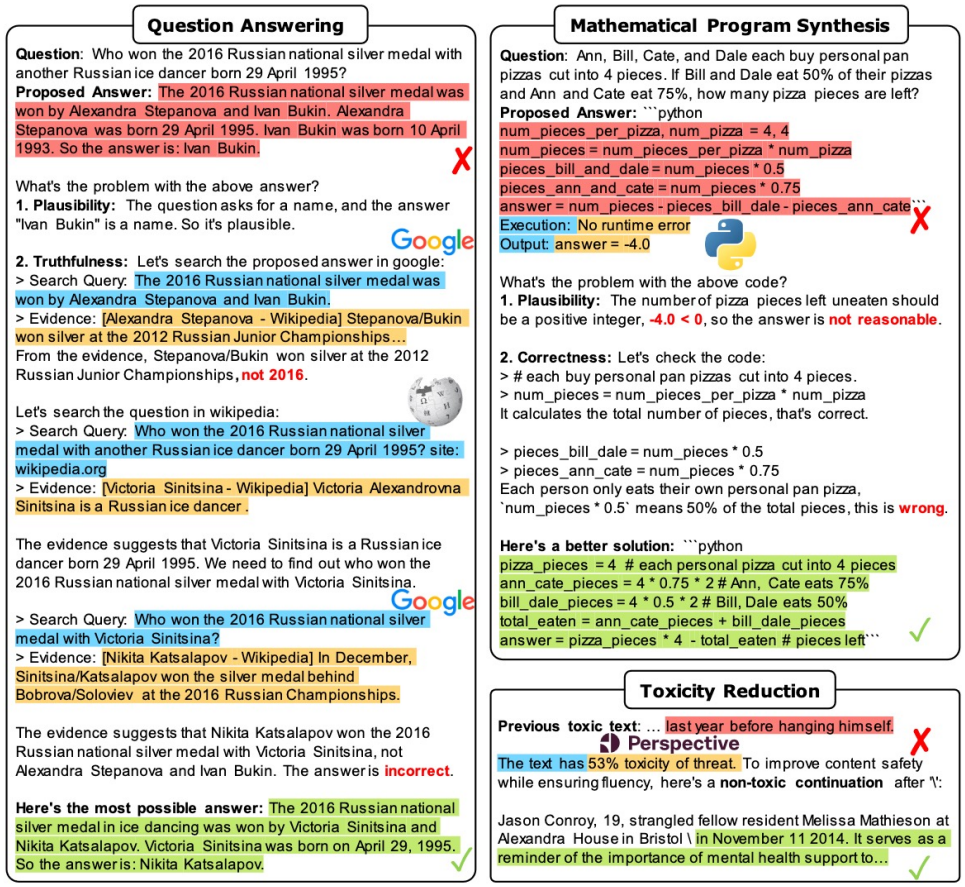
Critic



CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing - https://arxiv.org/abs/2305.11738

Critic

Methods	AmbigNQ		TriviaQA		HotpotQA	
	EM	F1	EM	F1	EM	F1
			Text-Davinci-003			
Vanilla	35.1	52.4	68.3	76.8	23.2	36.6
CoT	44.2	58.6	67.4	74.5	33.7	46.1
Self-Consistency	44.6	58.5	67.3	74.5	34.9	47.5
ReAct	47.6	61.2	64.4	71.6	34.9	47.9
ReAct → CRITIC	**51.4**	**66.2**	71.2	79.5	37.3	50.2
CRITIC	50.0	64.9	**72.7**	**80.6**	**38.7**	**50.5**
CRITIC w/o Tool	42.0	58.3	67.3	74.7	34.9	46.1
CRITIC*	**59.8**	**71.8**	**77.0**	**83.7**	**43.1**	**54.5**
Rejection Sampling	53.6	67.6	72.4	79.4	40.3	54.3
			ChatGPT (gpt-3.5-turbo)			
Vanilla	36.0	54.6	70.4	79.3	24.3	36.6
CoT	51.8	64.3	72.9	79.2	32.7	42.8
Self-Consistency	52.6	65.4	75.4	81.3	35.8	47.0
ReAct	52.0	64.8	63.7	69.8	39.1	50.2
ReAct → CRITIC	60.4	72.2	**75.5**	**81.8**	37.9	50.0
CRITIC	**62.0**	**74.9**	75.1	81.7	**40.3**	**52.9**
CRITIC w/o Tool	55.2	67.3	73.5	79.9	33.1	46.1
CRITIC*	**69.6**	**79.9**	80.9	86.6	**44.3**	**56.9**
Rejection Sampling	60.9	72.6	**82.0**	**87.1**	42.0	55.6
Supervised SoTA	-	52.1[a]	77.3[b]	-	67.5[c]	72.0[c]

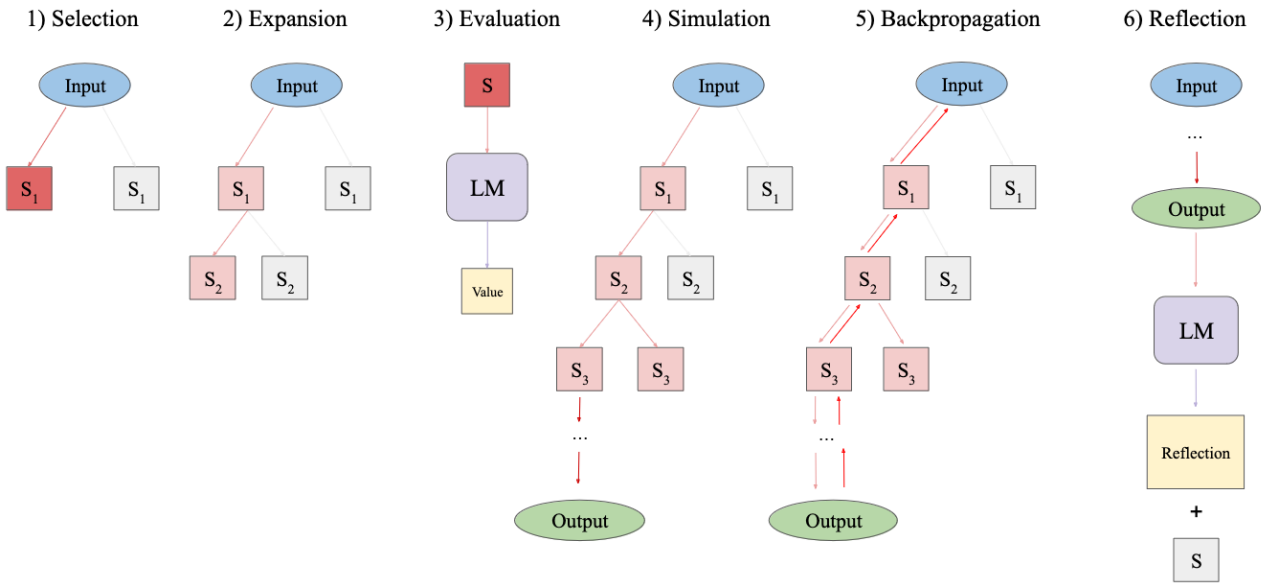


CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing - https://arxiv.org/abs/2305.11738

Exploration and Planning - MCTS

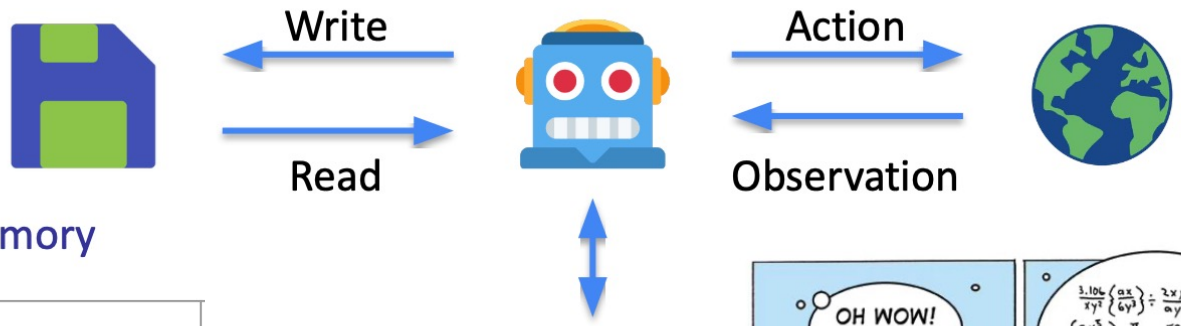
Prompt Method	HotpotQA (EM)
I/O	0.32
CoT (Wei et al., 2022)	0.34
CoT - SC (Wang et al., 2022)	0.38
ToT (Yao et al., 2023a)	0.55
RAP (Hao et al., 2023)	0.60
RAP (n = 10)	0.60
LATS (CoT)	**0.60**

Prompt Method	HotpotQA (EM)
ReAct (Yao et al., 2023b)	0.32
ReAct (best of k)	0.38
Reflexion (Shinn et al., 2023)	0.51
LATS	0.61
LATS (n = 3)	0.56
LATS (n = 10)	0.64
LATS (CoT + ReAct)	**0.71**



Language Agent Tree Search Unifies Reasoning Acting and Planning in Language Models - https://arxiv.org/abs/2310.04406

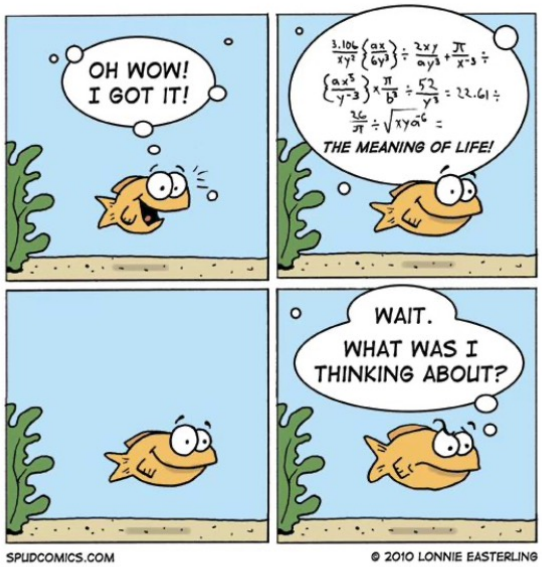
Memory



A long-term memory

Type by content	Definition
Episodic memory	Stores experience
Semantic memory	Stores knowledge
Procedural memory	Stores skills

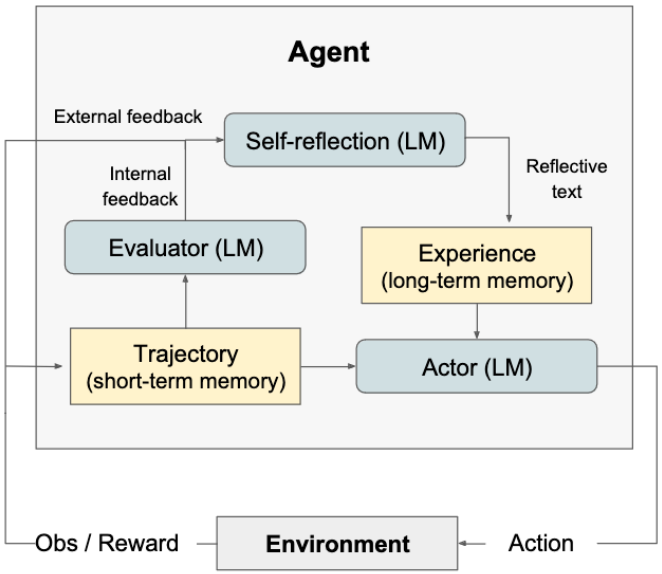
Instruction: …
Thought: …
Action: …
Obs: …
Thought: …
……

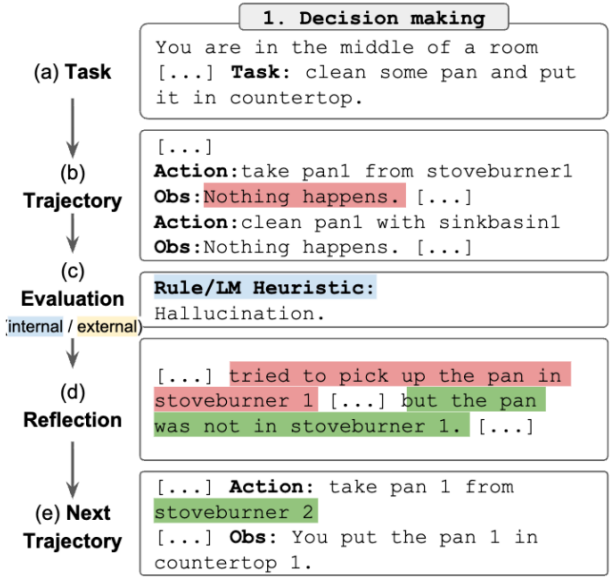


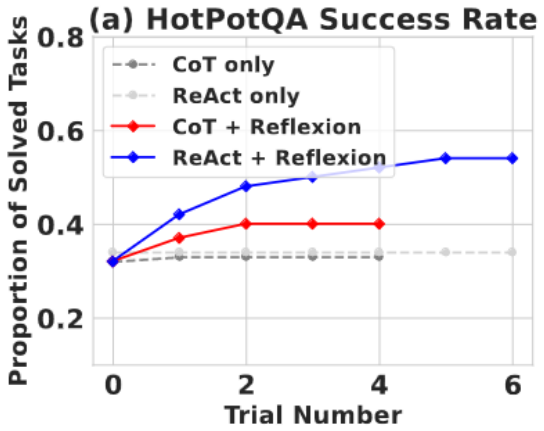
THE TRAGEDY OF A THREE SECOND MEMORY

Source: EMNLP 2024 Tutorial: Language Agents: Foundations, Prospects, and Risks https://languae-agent-tutorial.github.io

Reflexion (Agent with semantic memory)

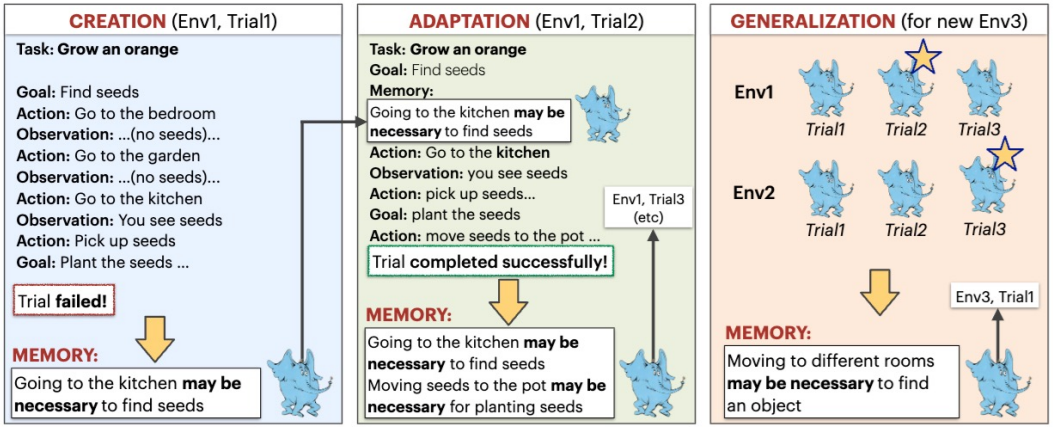


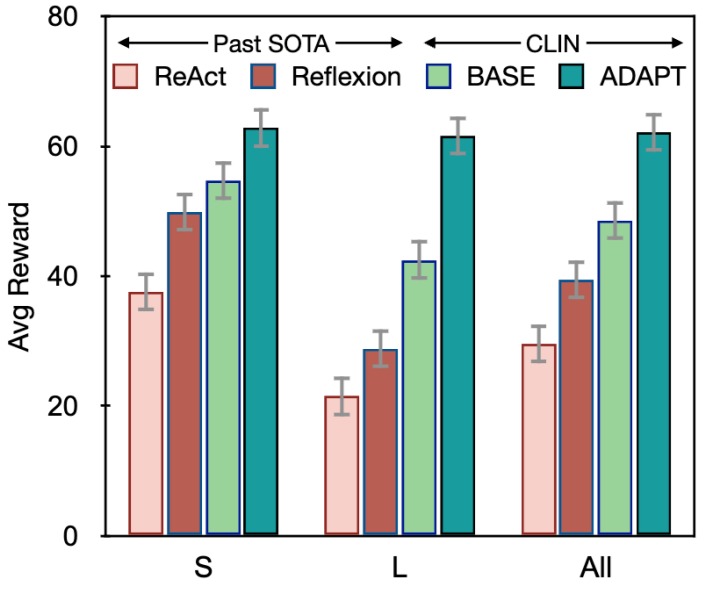




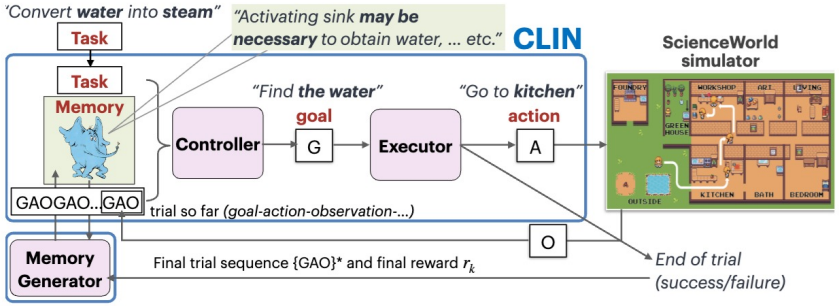
Reflexion: Language Agents with Verbal Reinforcement Learning - https://arxiv.org/abs/2303.11366

Causal (semantic) Memory





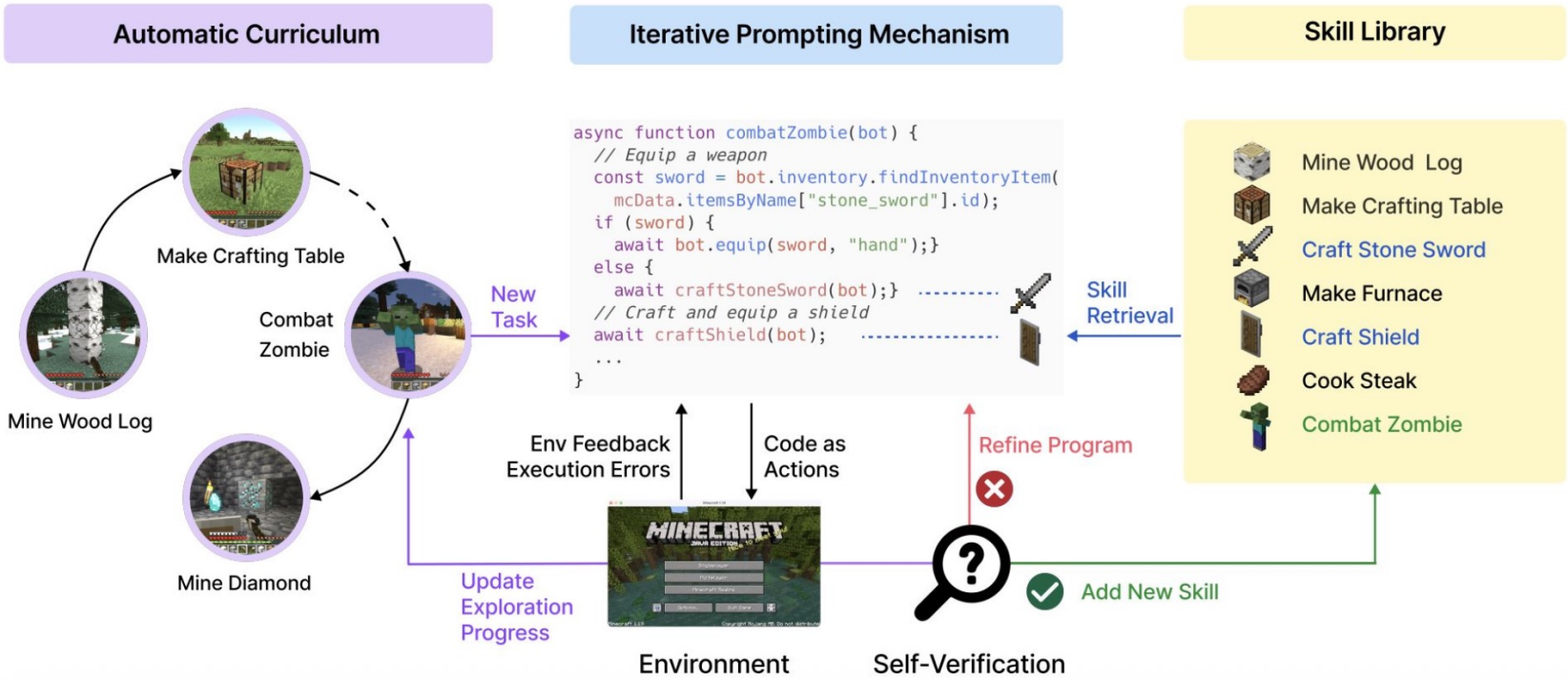

```
Each numbered item in the summary can ONLY be of the form:
X MAY BE NECCESSARY to Y.
X SHOULD BE NECCESSARY to Y.
X MAY BE CONTRIBUTE to Y.
X DOES NOT CONTRIBUTE to Y.
```



[Clin: A Continually Learning Language Agent For Rapid Task Adaptation And Generalization – Majmumder, et al. 2023]

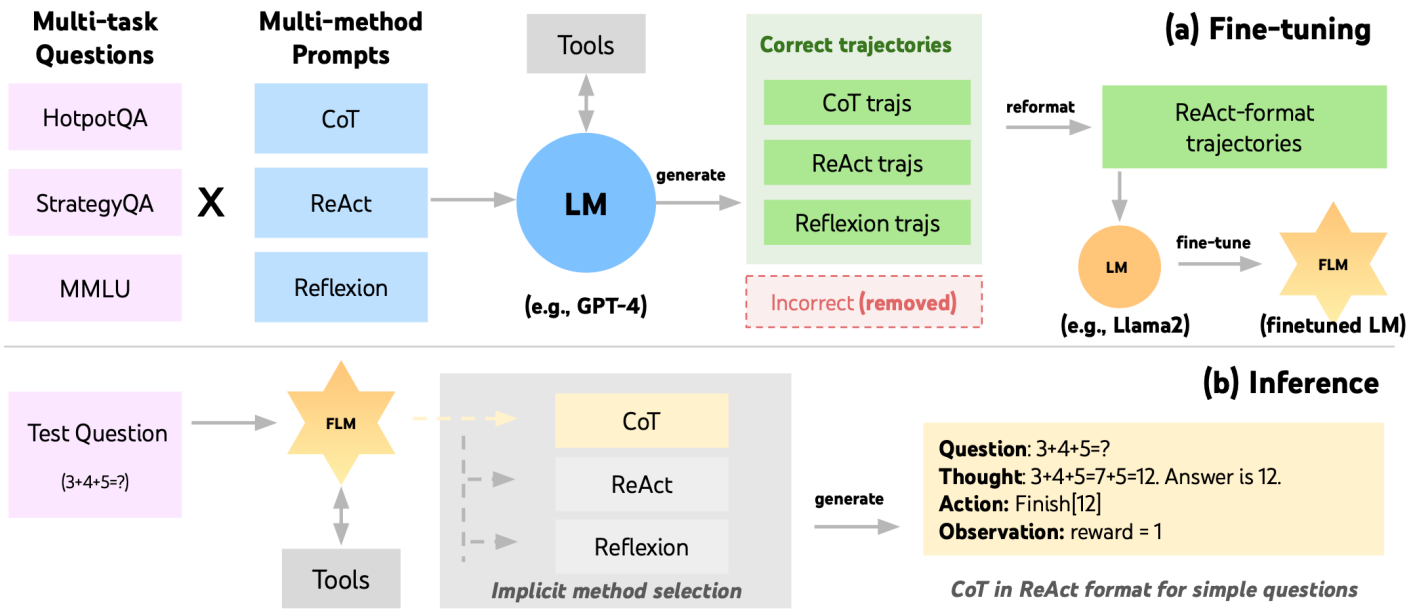
Voyager: Procedural memory

- Write: Code-based skills
- Read: Embedding retrieval



Source: EMNLP 2024 Tutorial: Language Agents: Foundations, Prospects, and Risks https://languae-agent-tutorial.github.io

FireAct (fine-tuning the agent)



- Create a collection of diverse tasks, and solve them using ReAct-like approaches
- Collect successful (~500 per task) trajectories from a more powerful LLM (i.e., GPT-4)
- Fine-tune the other LLMs on the data.

FireAct: Toward Language Agent Finetuning - https://arxiv.org/pdf/2310.05915

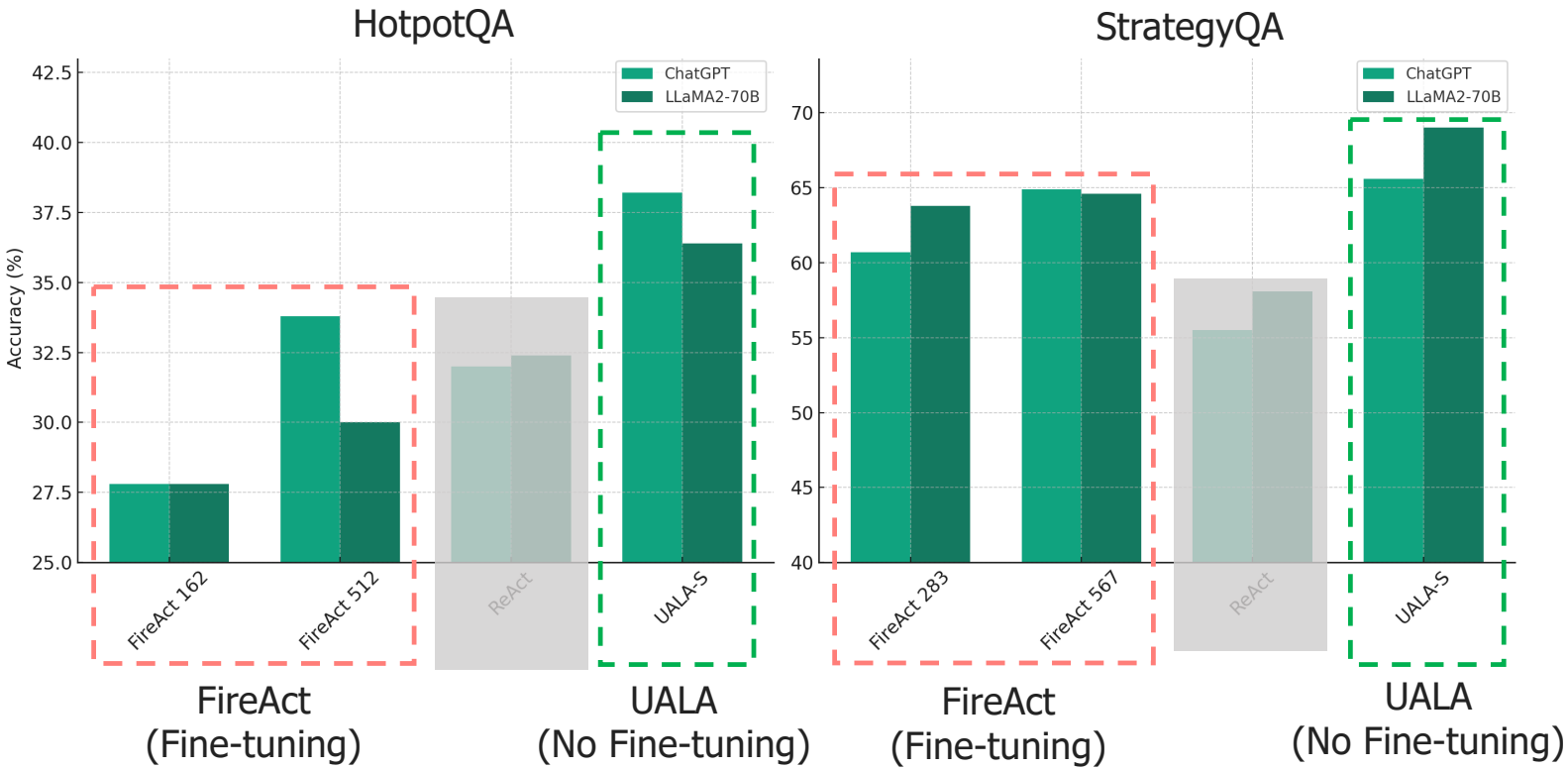
FireAct – snippet of results

- Fine-tuning significantly increases agent performance: Few-shot on ReAct prompting GPT-3.5 on HotpotQA achieves an exact match (EM) score of 31.4. Fine-tuning with 500 ReAct trajectories improves the EM to 39.2.
- Fine-tuning is cheaper and faster during agent inference
- Robustness to noisy tools: Improves performance by 64% in face of random tool output
- Generalization to new tasks (Bamboogle – MultiHop QA task designed to fail Google Search)

		ReAct	FireAct
Performance	HotpotQA	31.4	**39.2**
	MMLU	58.6	**62.4**
	GSM8K	53.4	**77.0**
	HumanEval	72.0	**74.5**
Cost	Money (10^{-3})	2.6	**2.2**
	Time (s)	9.0	**2.7**
Robustness	Normal	31.4	39.2
	None	20.8	33.6
	Random	22.6	37.2
Generalization	Bamboogle	40.8	44.0

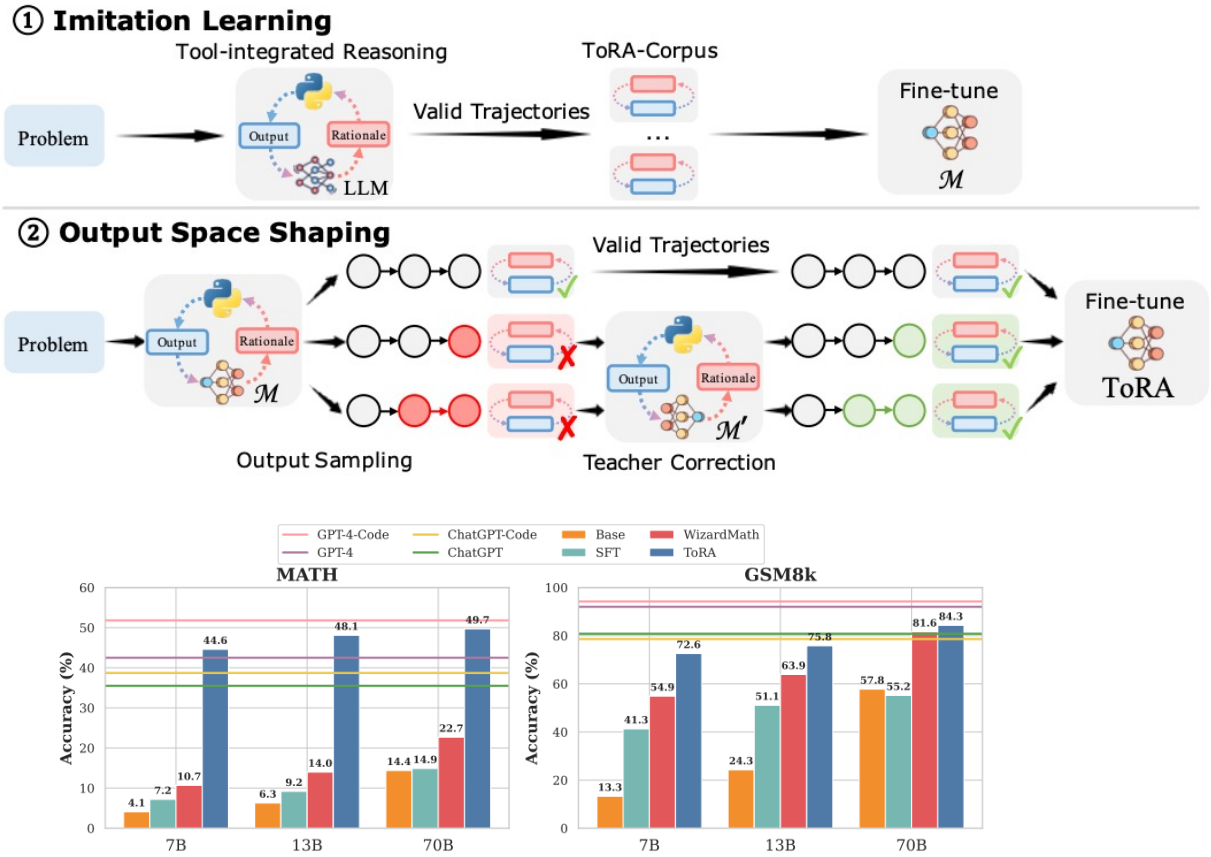
FireAct: Toward Language Agent Finetuning - https://arxiv.org/pdf/2310.05915

Small Data: UALA vs. FireAct



Towards Uncertainty-Aware Language Agent - https://arxiv.org/pdf/2401.14016

ToRA



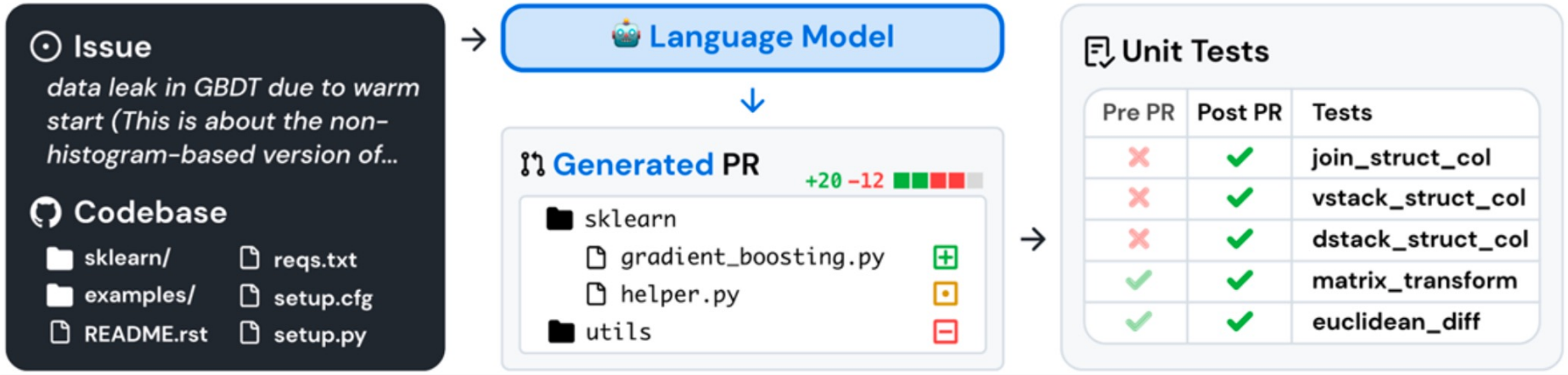
ToRA: A Tool-Integrated Reasoning Agent for Mathematical Problem Solving - https://arxiv.org/pdf/2309.17452

Environment

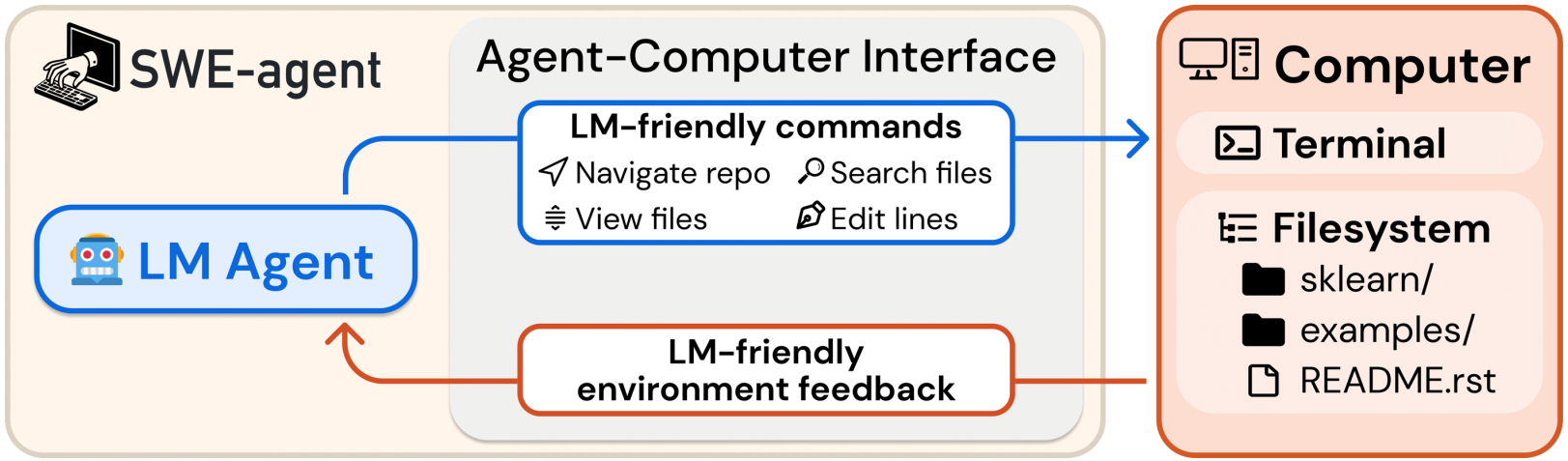
Digital world
Coding agents
Gaming agents
Mobile agents
Web/app agents
Computer agents
Physical world / Robotics

Coding Agents / SWE-Agent

- Environment: project code repos, filesystems, IDEs…
- Observation space: code files, exe outputs, docs, errors, commit history…
- Action space: code edits, file search/view, test updates…



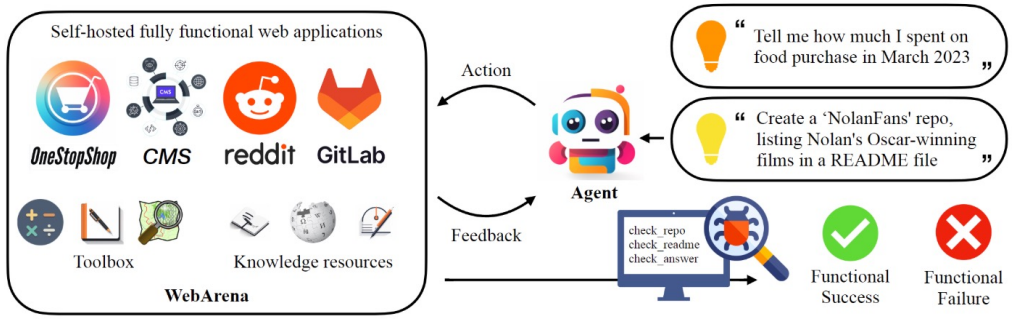
Coding Agents / SWE-Agent



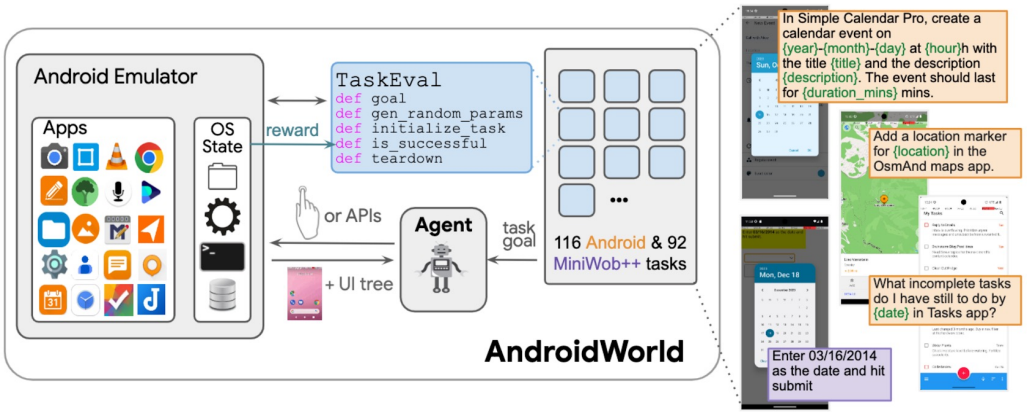
https://arxiv.org/pdf/2405.15793
https://www.swebench.com/

Prediction – "**Anthropic CEO: We might be 6-12 months away from when models do most maybe all of what Software engineering do end-to-end.**"

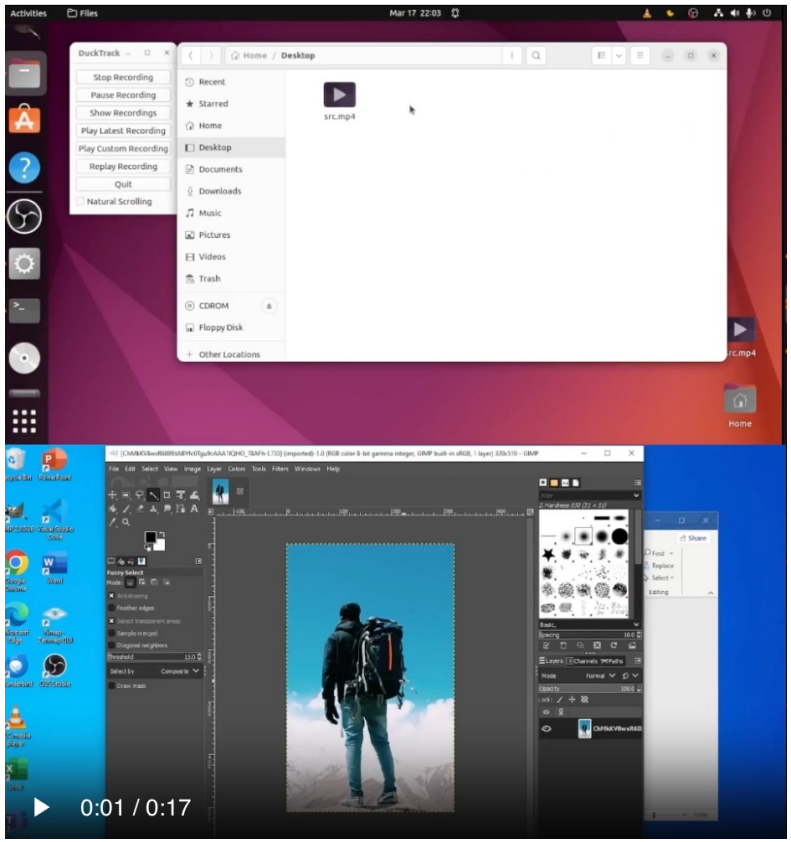
Android Agents / WebArena / OSWorld



https://webarena.dev



https://github.com/google-research/android_world



https://os-world.github.io/

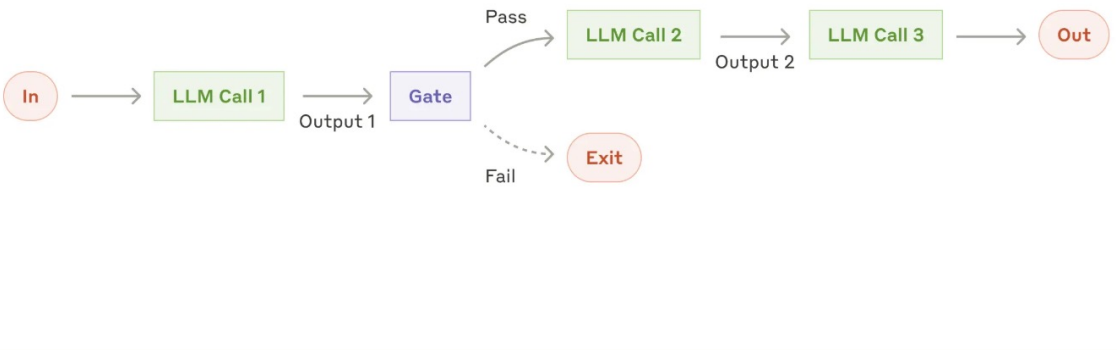
Pace of progress

1	10/2025	AGI-0	AI agent	✗	-	AGI-0	Screenshot	97.4
2	10/2025	askui AndroidVisionA	AI agent	✔	-	askui AndroidVisionA	Screenshot	94.8
2	01/2026	AutoDevice	AI agent	✔	-	gemini 3 pro + sonne	Screenshot	94.8
4	10/2025	DroidRun	AI agent	✔	-	GPT5, Gemini 2.5 Pr	Screenshot + A11y…	91.4
4	9/2025	mobile-use	AI agent	✔	-	Llama 4-scout, Gemi	Screenshot + A11y…	91.4
6	12/2025	Agent-Visco	AI agent	✗	-	-	Screenshot	88.8
7	10/2025	Surfer 2	AI agent	✗	-	o3 + holo1.5-72b	Screenshot	87.1
8	10/2025	gbox.ai	AI agent	✔	-	Sonnet 4.5 + Sonnet	Screenshot	86.2
9	8/2025	AutoGLM-Mobile	Model	✗	9B	AutoGLM-Mobile	Screenshot + A11y…	80.2
10	9/2025	LX-GUIAgent	AI agent	✗	-	LX-GUIAgent	Screenshot + A11y…	79.3
11	12/2025	AgentProg	AI agent	✔	-	Gemini-2.5-Pro+UI-T	Screenshot	78.0
12	8/2025	Finalrun	AI agent	✗	-	GPT-5	Screenshot + A11y…	76.7
12	9/2025	K²-Agent	AI agent	✗	72B + 7B	Qwen2.5-VL-72B + C	Screenshot	76.7
12	1/2026	MAI-UI	Model	✗	235B	MAI-UI-235B-A22B	Screenshot	76.7
15	9/2025	MobileUse-v2	AI agent	✔	32B	Hammer-UI-32B	Screenshot	75.0
16	8/2025	Mobile-Agent-v3	AI agent	✗	32B	GUI-Owl-32B	Screenshot	73.3
16	1/2026	MAI-UI	Model	✗	32B	MAI-UI-32B	Screenshot	73.3
18	1/2026	MAI-UI	Model	✔	8B	MAI-UI-8B	Screenshot	70.7
19	10/2025	Gemini 2.5 Compute	Model	✗	-	Gemini 2.5 Computer	Screenshot	69.7
20	6/2025	JT-GUIAgent-V2	AI agent	✗	-	JT-GUIAgent-V2	Screenshot	67.2
21	8/2025	GUI-Owl-7B	Model	✗	7B	GUI-Owl-7B	Screenshot	66.4
22	8/2025	UI-Venus	Model	✔	72B	UI-Venus-Navi-72B	Screenshot	65.9
23	07/2025	MobileUse	AI agent	✔	72B	Qwen2.5-VL-72B	Screenshot	62.9
24	05/2025	Seed1.5-VL	Model	✔	20.B	Seed1.5-VL	Screenshot + A11y…	62.1
25	6/2025	JT-GUIAgent-V1	AI agent	✗	-	JT-GUIAgent-V1	Screenshot	60.0
26	3/2025	V-Droid Paper	AI agent	✔	8B	V-Droid (Llama8B)	A11y tree	59.5
27	4/2025	Agent S2	AI agent	✔	-	Agent S2	Screenshot	54.3
28	8/2025	UI-Venus	Model	✔	7B	Venus-Navi-7B	Screenshot	49.1
28	1/2026	MAI-UI	Model	✔	2B	MAI-UI-2B	Screenshot	49.1
30	05/2025	GUI-Explorer	AI agent	✔	-	GPT-4o	Screenshot + A11y…	47.4
31	4/2025	AndroidGen	AI agent	✔	-	GPT-4o	A11y tree	46.8
32	1/2025	UI-TARS	Model	✔	72B	UI-TARS	Screenshot	46.6
33	12/2024	Aria-UI	Model	✔	-	GPT-4o + Aria-UI	Screenshot	44.8
34	4/2025	ScaleTrack	Model	✗	8B	ScaleTrack-7B	A11y tree	44.0
34	1/2025	UGround	Model	✔	-	GPT-4o + UGround	Screenshot	44.0

Workflow examples

Workflow: Prompt chaining

Prompt chaining decomposes a task into a sequence of steps, where each LLM call processes the output of the previous one. You can add programmatic checks (see "gate" in the diagram below) on any intermediate steps to ensure that the process is still on track.



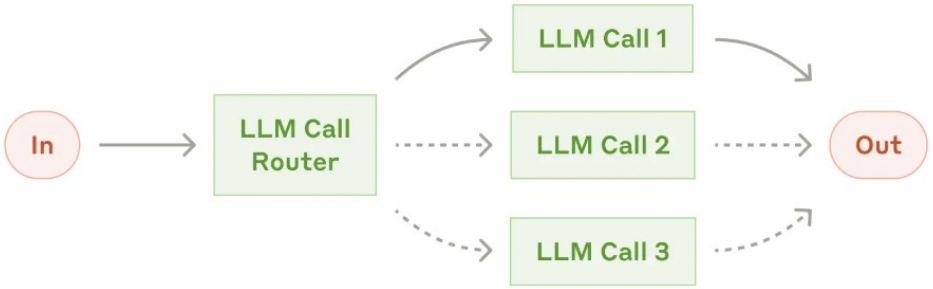
The prompt chaining workflow

https://www.anthropic.com/engineering/building-effective-agents

Workflow examples

Workflow: Routing

Routing classifies an input and directs it to a specialized followup task. This workflow allows for separation of concerns, and building more specialized prompts. Without this workflow, optimizing for one kind of input can hurt performance on other inputs.



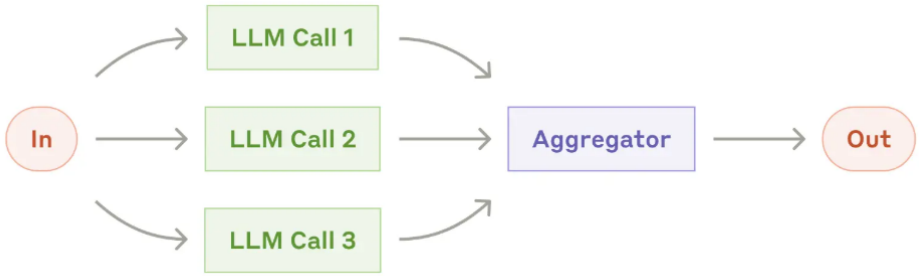
https://www.anthropic.com/engineering/building-effective-agents

Workflow examples

Workflow: Parallelization

LLMs can sometimes work simultaneously on a task and have their outputs aggregated programmatically. This workflow, parallelization, manifests in two key variations:

- **Sectioning:** Breaking a task into independent subtasks run in parallel.
- **Voting:** Running the same task multiple times to get diverse outputs.



https://www.anthropic.com/engineering/building-effective-agents

Workflow examples

Workflow: Orchestrator-workers

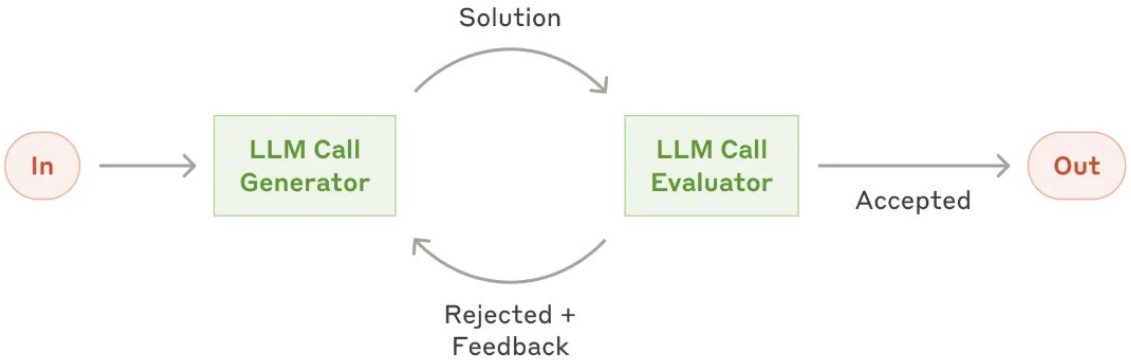
In the orchestrator-workers workflow, a central LLM dynamically breaks down tasks, delegates them to worker LLMs, and synthesizes their results.

https://www.anthropic.com/engineering/building-effective-agents

Workflow examples

Workflow: Evaluator-optimizer

In the evaluator-optimizer workflow, one LLM call generates a response while another provides evaluation and feedback in a loop.



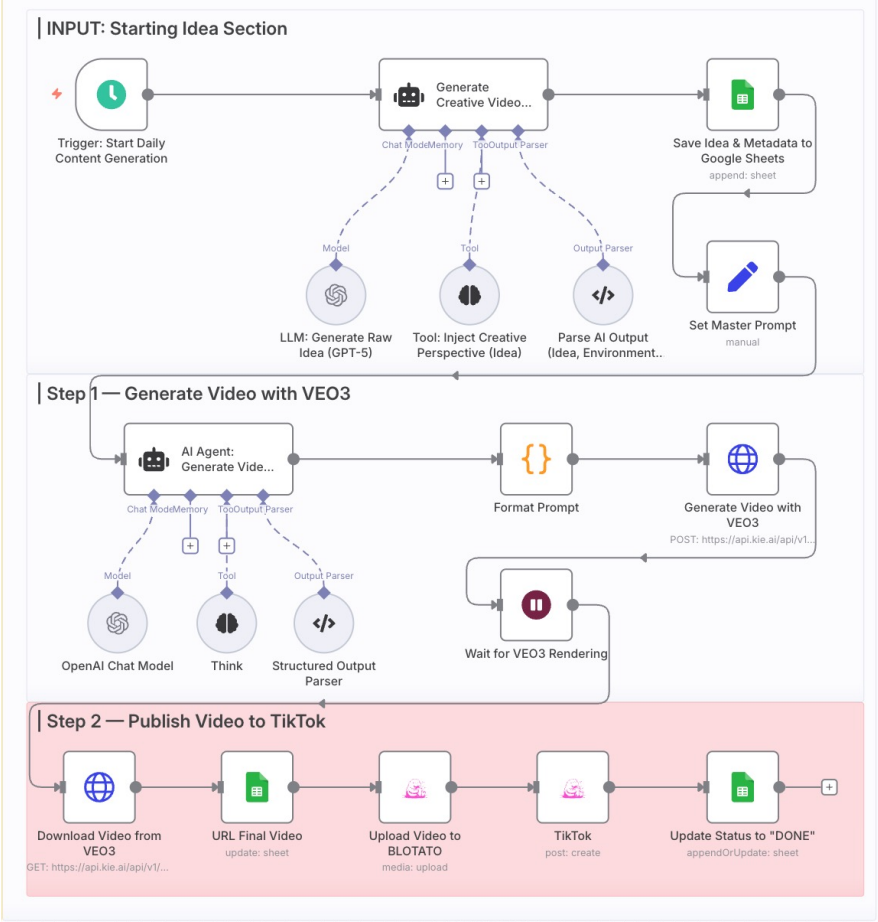
https://www.anthropic.com/engineering/building-effective-agents

Feel like creating your own agentic workflow?

https://github.com/n8n-io/n8n

Can setup one with a few hours of effort to run endlessly on your laptop for free, using an offline open-source LLM!

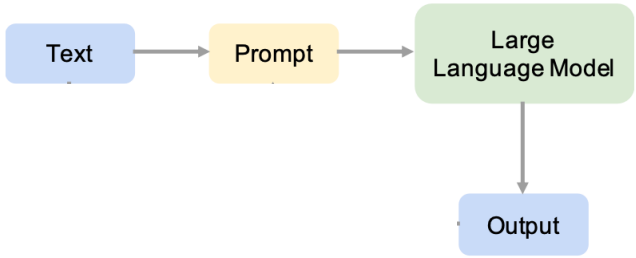
It is almost a no-code framework.



Separately, you can also build your own agent with a few lines of code: https://minimal-agent.com/#our-first-prototype-in-50-lines

Can we build a verifier for LLMs' structured outputs?

Standard

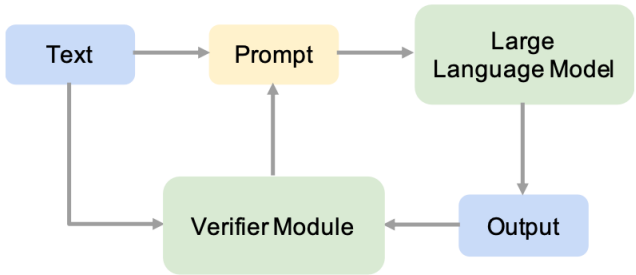


Question: How to improve the LLM's performance without fine-tuning?

An answer: Train a smaller LM to operate as a verifier for the LLM and to provide fine-grained feedback to the LLM.

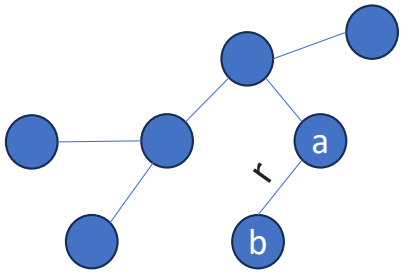
To train such a verifier, we need the data. Which we do not have. Will see next how this is constructed. Once the data is constructed, we train a T5 model as the verifier.

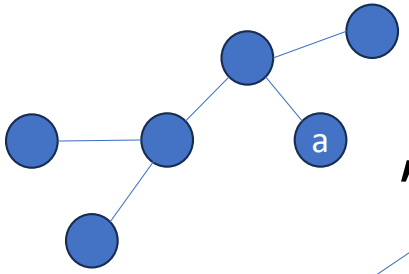
PiVe



PiVe: Prompting with Iterative Verification to Improve Graph-based Generative Capability of LLMs, Han, et al. 2024

Verifier Training Data Preparation

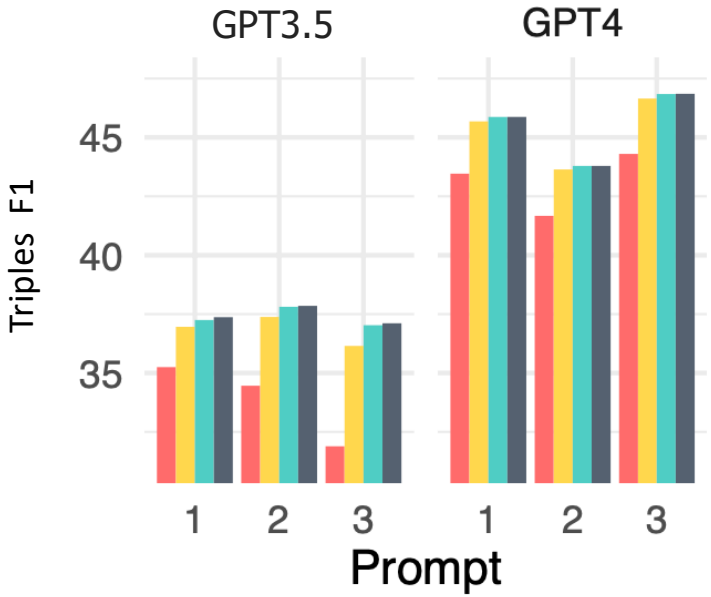
(Text, G: 

) Perturb G (e.g., remove [a,r,b]) (Text, $\overline{G}$: 

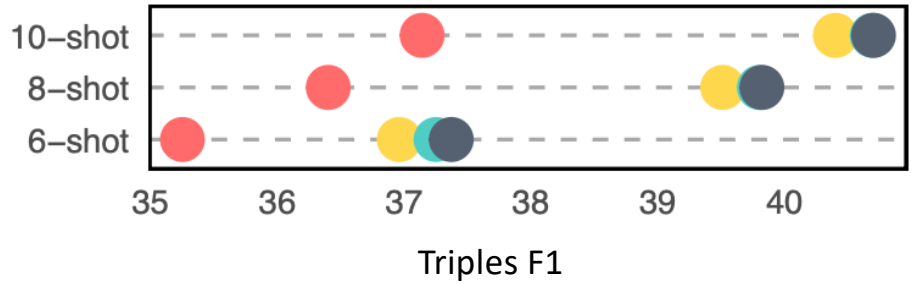
, [a,r,b])

Verifier Training Data = Subset of (T,G, "Correct") pairs from a seed T2G dataset (i.e., given T, G to generate "Correct"). + Set of (T, $\overline{G}$, Feedback) Triples for training verifier (i.e., given T, $\overline{G}$ to generate Feedback).

Snippet of results – More shots, more gain



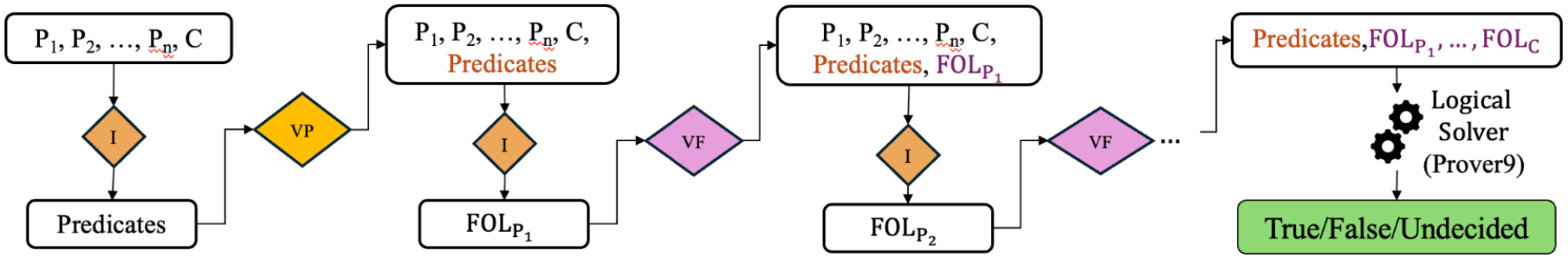
The colors represent Base, and corrective iterations 1, 2, 3.



Results are On KELM T2G dataset (Agarwal et al., NAACL 2021)

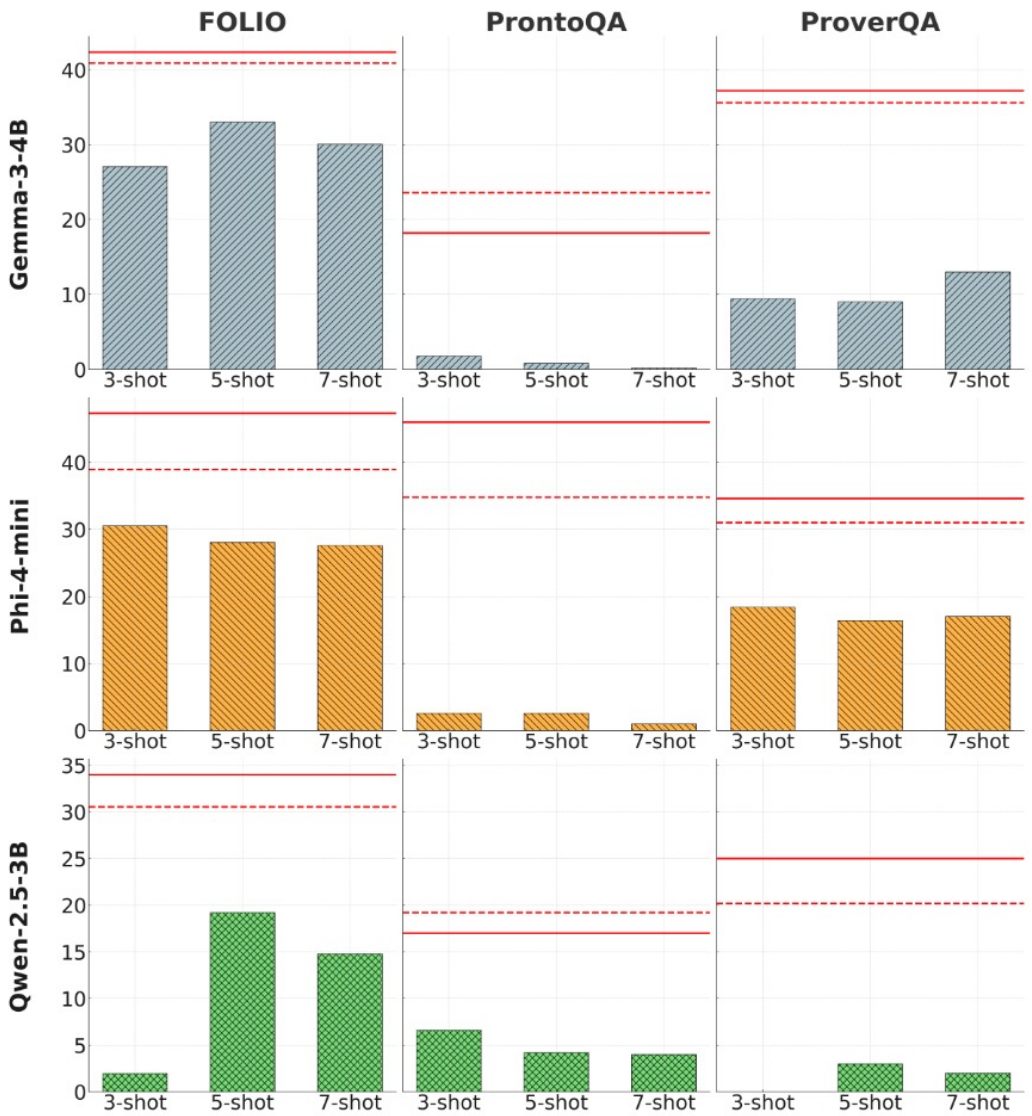
Prompt 1: Transform the text into a semantic graph.
Prompt 2: Transform the text into a semantic graph consisting of a set of triples. Generate as many triples as possible.
Prompt 3: Transform the text into a semantic graph consisting of a set of triples. First produce all relations possible, then produce the graph.

Tool-augmented reasoning improves results.
Can we build translation verifier for tools? i.e. FOLs



Generated predicates and FOLs are passed to a verifier (a specialised T5 model) to either confirm correctness or to apply the required error corrections before the next step of the translation.

Integration of verifier (denoted by solid red line), enhances reasoning accuracy substantially



Can agents reason in the wild?

- LLMs are good at solving well-defined reasoning problems, but real-world scenarios are complex and challenging
- We introduce the task of reasoning in the wild: LLMs to solve problems of hybrid types by decomposing them into subproblems and applying corresponding formalisms/tools to solve them step by step.
- Created problems that are (1) ambiguous; and (2) requires multiple tactics to solve
  - Take existing problems (e.g., GSM8K, FOLIO, ReClor) and "blend" them into a hybrid problem

Ambiguous, Mixed-in-scope Problem

Context:

Jerry works at Acme Corporation, whose strategic decisions are set to significantly affect its employees' daily routines. Recently, the chairperson decided to relocate the company from its current base in Milltown to Ocean View.

...

To celebrate the relocation, Jerry helped preparing candy gift bags. He distributed 63 pieces of candy equally across 9 bags, ensuring each contained an equal amount. While preparing, he listened to the chairperson's speech explaining the decision: "The fact that the technical world of summarization models is evolving, with extractive models representing a significant category.

...

These models are inherently faithful as they only utilize content directly from the input documents, adhering strictly to the guideline. That's why we need to move", said the chairperson.

...

Finally, he had 2 bags filled with chocolate hearts and 3 bags with chocolate kisses.

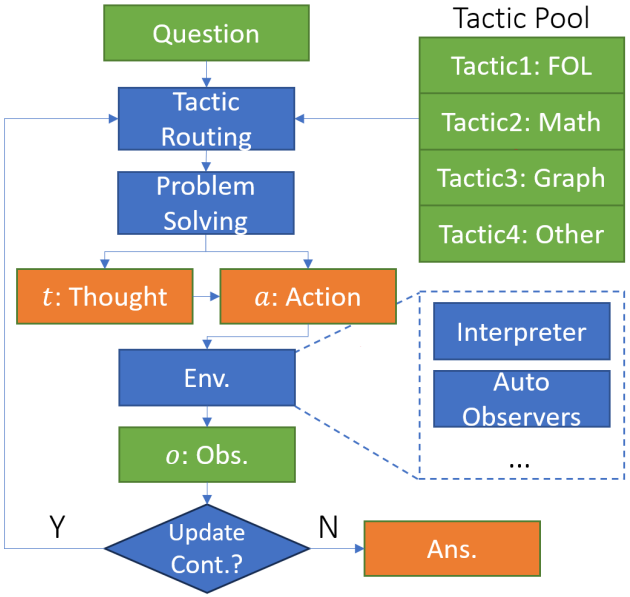
Which one of the following is correct:

1. The number of candies Jerry put that were not chocolate is 26.
2. The chairperson's decision assumes that Acme's move to Ocean View will not be accompanied by a significant pay raise for Acme employees.
3. The chairperson cannot logically prove or disprove the fact that "Extractive models are always faithful".

Can LLMs Reason in the Wild with Programs? Yang, etl. 2024

Reasoning in the wild – Inference Flow

- **Tactic**: a specific formalism defined for a problem type
- **Routing**: given a problem of unknown type, the agent chooses the right tactic to approach it
- **Execution**: the agent performs T thought, A action, and receives O observation at every round
- **Iteration**: the agent iteratively identifies the subproblems, solve them, and produce the final answer



Tactic

Tactic name: predicate_logic_z3
Problem type and tactic: This tactic builds a formal logical model using predicate logic using z3 lib …
Code template:

```
import z3
def check_model(solver):
    …
def main():
    s = z3.Solver()
    <your code>
```

Action Space:
#A# Plan
Input: the problem given
Functionality: give a plan for the question, include a sketch of the solution and libs to be used
Output: text description of the plan, code snippets

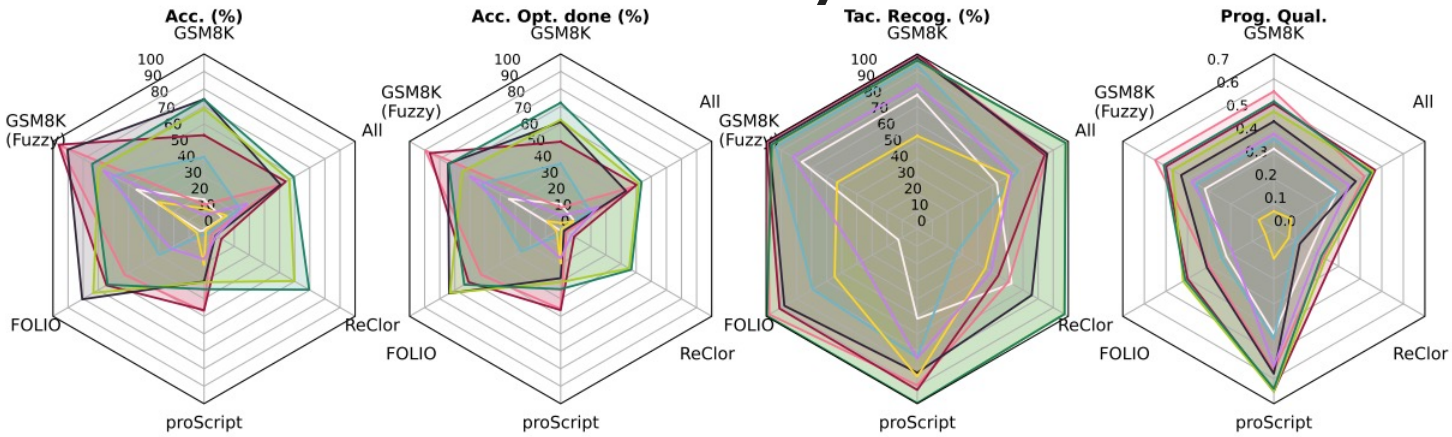
#A# Build FOL model
Input: the problem given
Functionality: build the FOL system …
Output: the main() function with z3 code

#A# Revise code
…
#A# Aggregate and answer
…

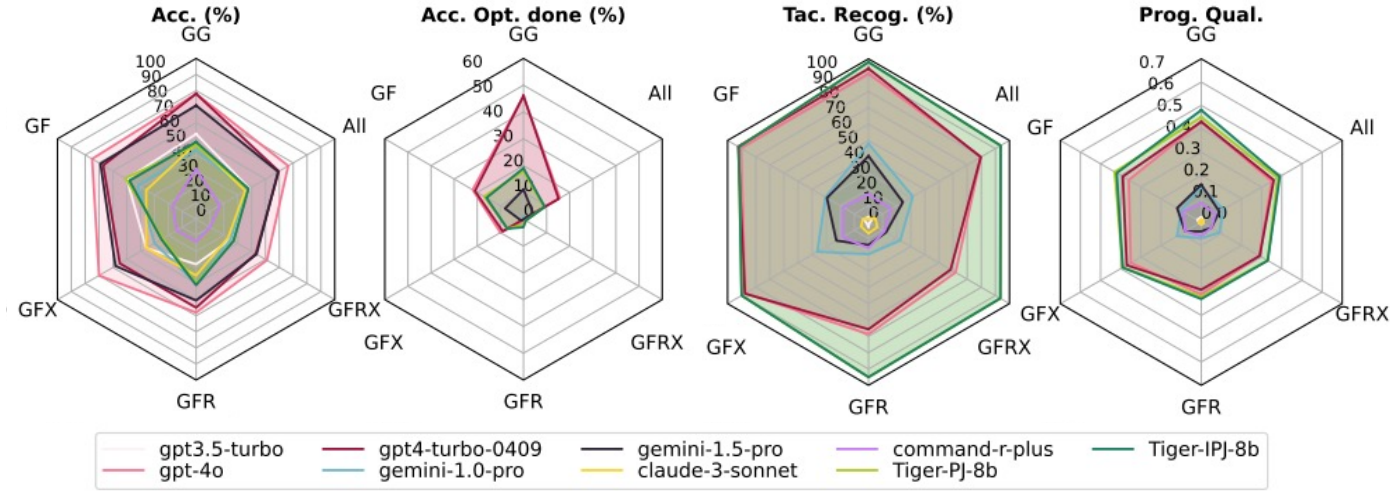
example of FOL tactic

Experiments – Standalone vs. Hybrid

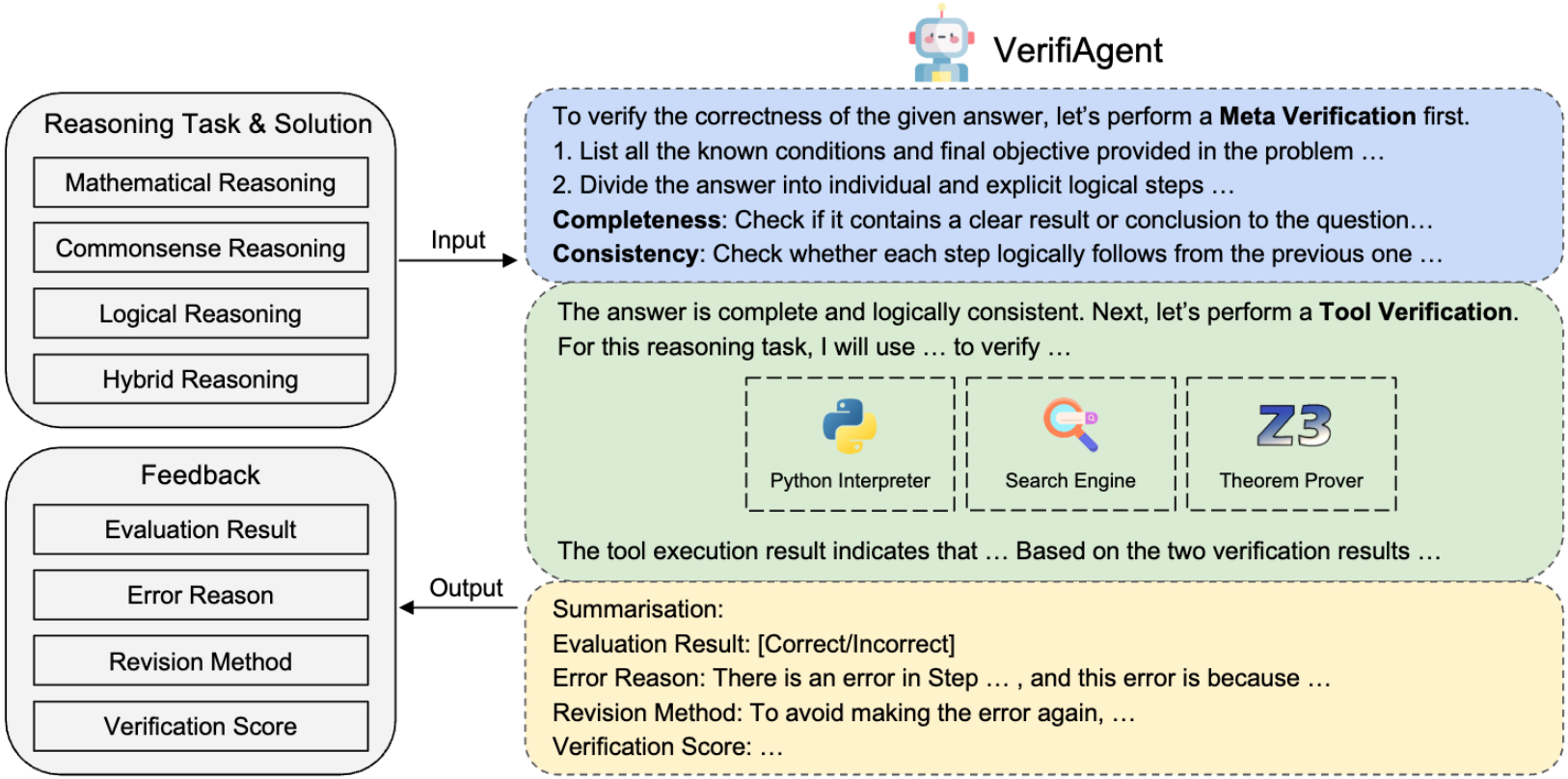
Performance on Standalone Problems



Performance on Hybrid Problems

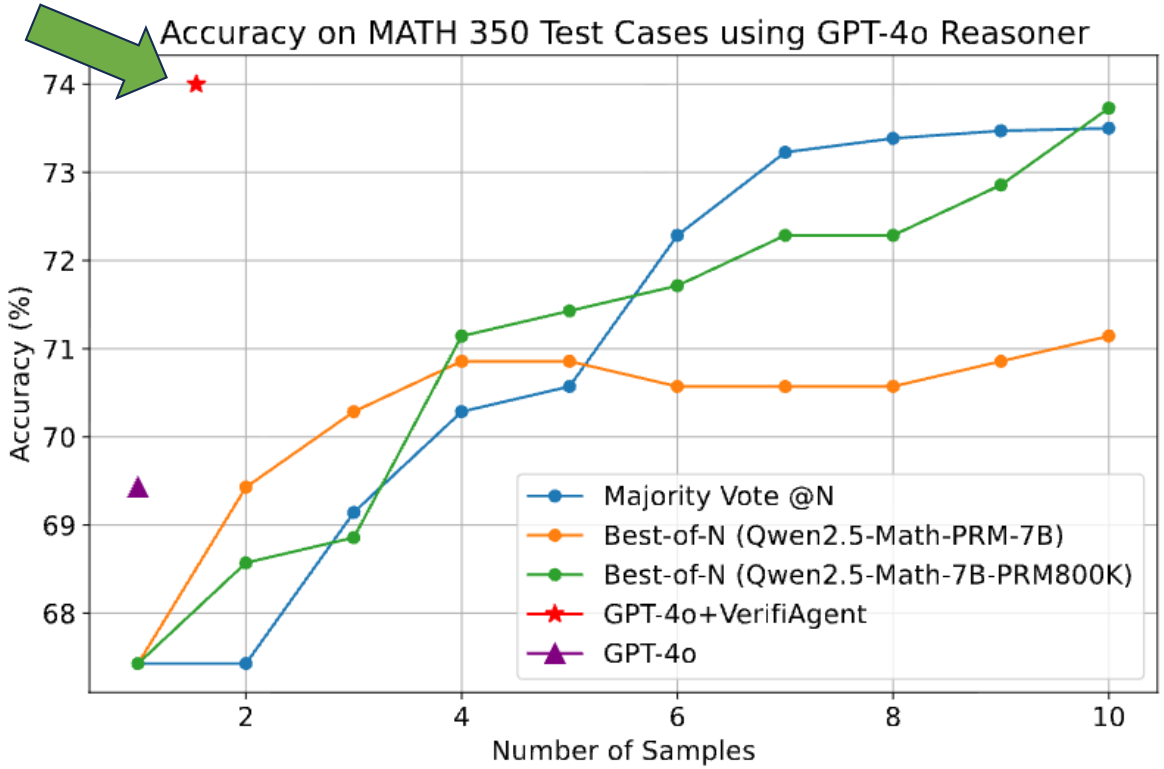


VerifiAgent (unifying verification methods)



https://aclanthology.org/2025.findings-emnlp.891.pdf

Better results, with less number of samples



https://aclanthology.org/2025.findings-emnlp.891.pdf

Accelerating Scientific Research with Gemini: Case Studies and Common Techniques

Feb 2026

… researchers have successfully collaborated with advanced AI models … to solve open problems, refute conjectures, and generate new proofs across diverse areas in theoretical computer science, as well as other areas such as economics, optimization, and physics.

… we extract common techniques for effective human-AI collaboration in theoretical research, such as iterative refinement, problem decomposition, and cross-disciplinary knowledge transfer.

… deploying the model as a rigorous adversarial reviewer to detect subtle flaws in existing proofs, and embedding it within a "neuro-symbolic" loop that autonomously writes and executes code to verify complex derivations.

Finding a Fatal Flaw in a Cryptography Paper

A paper claimed a major breakthrough in cryptography

Method

- Researchers used Gemini as an *adversarial reviewer*
- Not just "check this proof," but a strict protocol:
  - Produce review
  - Critique its own review
  - Iterate for rigor
  - Explicitly flag gaps vs. proven claims

What the model found

A mismatch between:

- **Definition:** required *perfect consistency*
- **Delivered in paper:** only gave *statistical consistency*

This gap breaks the soundness argument.

Outcome

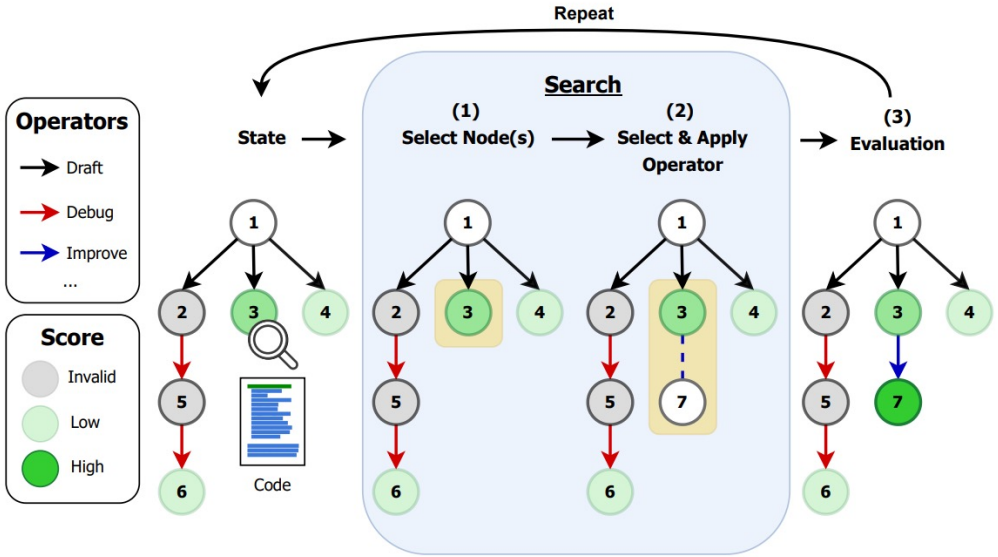
Human cryptography experts verified the issue
Authors acknowledged the flaw and updated the paper
The result undermined the main theorem

https://arxiv.org/pdf/2602.03837

AI Research Agents for Machine Learning: Search, Exploration, and Generalization in MLE-bench

Edan Toledo[1,2,*], **Karen Hambardzumyan**[1,2,*], **Martin Josifoski**[1,*], **Rishi Hazra**[3,†], **Nicolas Baldwin**[1], **Alexis Audran-Reiss**[1], **Michael Kuchnik**[1], **Parth Pathak**[1], **Despoina Magka**[1], **Minqi Jiang**[1], **Alisia Maria Lupidi**[1], **Andrei Lupu**[1], **Roberta Raileanu**[1], **Kelvin Niu**[1], **Tatiana Shavrina**[1], **Jean-Christophe Gagnon-Audet**[1], **Michael Shvartsman**[1], **Shagun Sodhani**[1], **Alexander H. Miller**[1], **Abhishek Charnalia**[1], **Derek Dunfield**[1], **Carole-Jean Wu**[1], **Pontus Stenetorp**[2], **Nicola Cancedda**[1], **Jakob Nicolaus Foerster**[1], **Yoram Bachrach**[1]

[1]FAIR at Meta, [2]University College London, [3]Örebro University
[*]Equal contribution (author order determined by a game of UNO), [†]Work done while at Meta



Operators are LLM-powered prompts:

Draft: "Here's the task, data overview, previously tried ideas... generate a NEW solution idea and complete implementation"

Improve: "Here's existing code and its output... propose ONE improvement and implement it"

Debug: "Here's buggy code and the error traceback... fix it without changing core idea"

Crossover: "Here are two solutions... combine their best aspects"

Memory: Extracts summaries of past attempts to inform future operators

https://arxiv.org/pdf/2507.02554

Example of MLE-bench: **"spooky-author-identification"** from Kaggle (2017)

Description:

Goal: Classify excerpts of horror fiction text by which famous author wrote them (likely authors like Edgar Allan Poe, Mary Shelley, HP Lovecraft, etc.)

Type: Multi-class text classification problem

Data: The agents have access to training text samples with author labels and must predict authors for unlabeled test excerpts

What the Agent Must Do

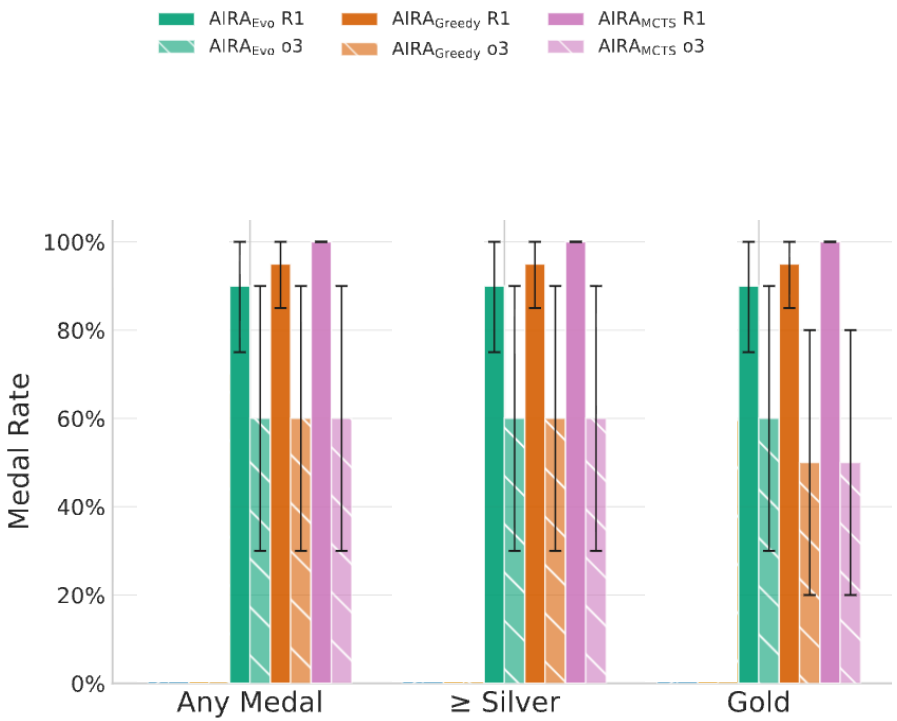
1. **Understand the problem** from the Kaggle competition description
2. **Explore the data** programmatically
3. **Design a solution approach** - might involve:
    1. Text preprocessing and feature engineering
    2. Selecting appropriate models (traditional ML like TF-IDF + classifier, or deep learning like BERT)
    3. Setting up cross-validation
4. **Implement the solution** as executable Python code
5. **Train and evaluate** the model using 5-fold cross-validation
6. **Generate predictions** and save them in the required submission.csv format

7. **Iteratively improve** the solution through debugging and refinement

Example Solution Path

A typical agent trajectory might:
1. Start with a simple baseline (e.g., TF-IDF + Logistic Regression)
2.Try different feature engineering approaches
3.Experiment with neural models or ensemble
4.Debug any errors in data processing or model training
5.Tune hyperparameters to maximize cross-validation score

The agents operate autonomously for 24 hours with access to 1 H200 GPU, 24 CPUs, and can install any needed Python packages.



Gold means "beat ~90% of human Kagglers"