Scientific Agents

Rithik Kapoor

TEXAS A&M UNIVERSITY
Department of Computer
Science & Engineering

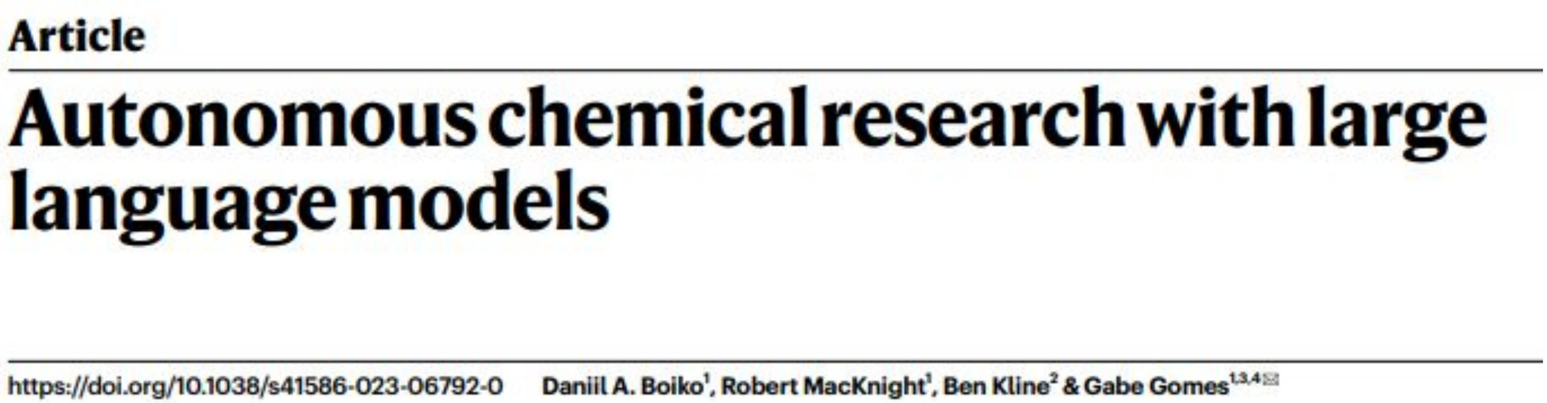
LLM Agents

- Traditional LLMs lack memory, struggle to plan multi-step tasks, and can’t interact dynamically with external tools or data sources.
- Agentic LLMs overcome these limitations by integrating planning, decision-making, memory, and tool-use capabilities, allowing them to execute complex, multi-step tasks autonomously.
- LLM Agents can autonomously research the internet, write and debug code, summarize documents, and dynamically adjust their actions based on intermediate results and real-time feedback.

Agenda

- Paper 1: Autonomous Chemical Research with Large Language Models
- Paper 2: Augmenting Large Language Models with Chemistry Tools
- Paper 3: Monte Carlo Thought Search: Large Language Model Querying for Complex Scientific Reasoning in Catalyst Design

Paper 1



Article

Autonomous chemical research with large language models

https://doi.org/10.1038/s41586-023-06792-0

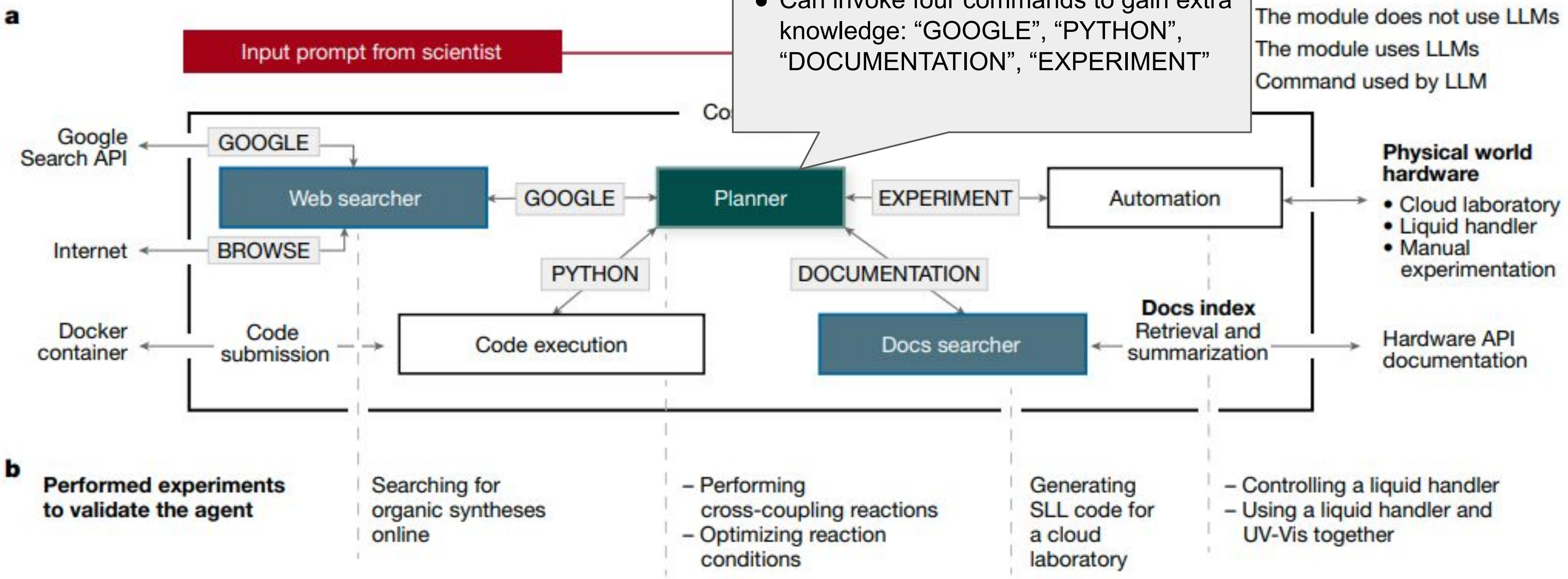
Daniil A. Boiko[1], Robert MacKnight[1], Ben Kline[2] & Gabe Gomes[1,3,4] ✉

Paper's Contributions

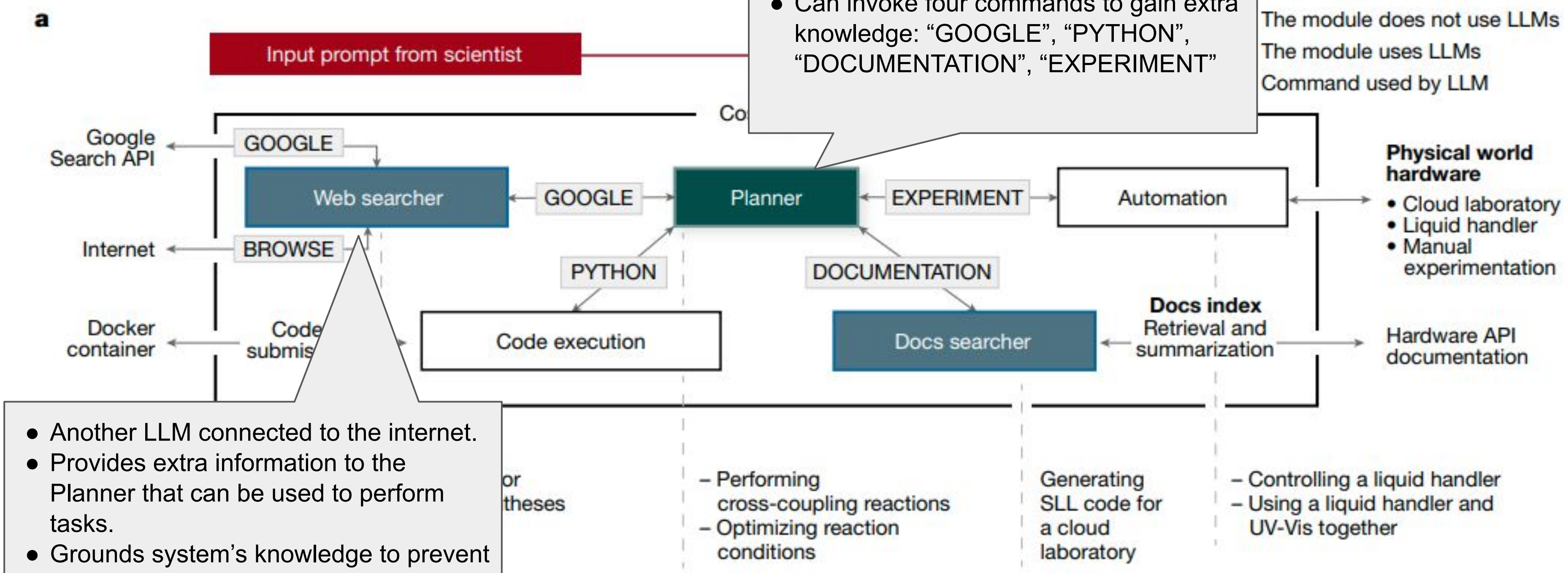
- Introduce CoScientist, an LLM based agent capable of autonomous design, planning, and performance of scientific experiments.
- Inference based method that uses tools to augment pretrained LLMs.
- Applied on 6 tasks:
    1. Planning chemical syntheses of known compounds using publicly available data
    2. Efficiently searching and navigating through extensive hardware documentation
    3. Using documentation to execute high-level commands in a cloud laboratory
    4. Precisely controlling liquid handling instruments with low-level instructions
    5. Tackling complex scientific tasks that demand simultaneous use of multiple hardware modules and integration of diverse data sources
    6. Solving optimization problems requiring analyses of previously collected experimental data

Overview of CoScientist

- Pretained LLM like GPT-4.
- Responsible for coordinating tasks to complete task provided by input prompt
- Can invoke four commands to gain extra knowledge: “GOOGLE”, “PYTHON”, “DOCUMENTATION”, “EXPERIMENT”


a
Input prompt from scientist
The module does not use LLMs
The module uses LLMs
Command used by LLM
Co
Google Search API
GOOGLE
Web searcher
GOOGLE
Planner
EXPERIMENT
Automation
Physical world hardware
• Cloud laboratory
• Liquid handler
• Manual experimentation
Internet
BROWSE
PYTHON
DOCUMENTATION
Docs index
Retrieval and summarization
Docker container
Code submission
Code execution
Docs searcher
Hardware API documentation
b
Performed experiments to validate the agent
Searching for organic syntheses online
– Performing cross-coupling reactions
– Optimizing reaction conditions
Generating SLL code for a cloud laboratory
– Controlling a liquid handler
– Using a liquid handler and UV-Vis together

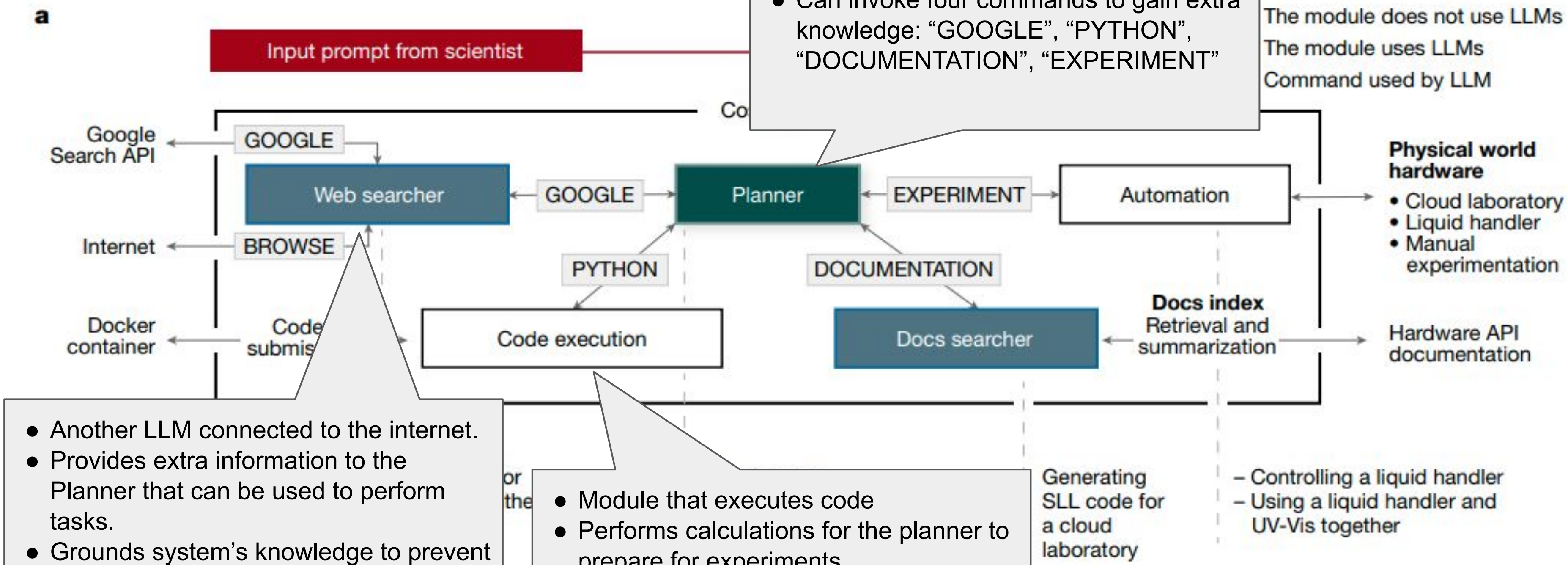
Overview of CoScientist



- Pretained LLM like GPT-4.
- Responsible for coordinating tasks to complete task provided by input prompt
- Can invoke four commands to gain extra knowledge: “GOOGLE”, “PYTHON”, “DOCUMENTATION”, “EXPERIMENT”

- Another LLM connected to the internet.
- Provides extra information to the Planner that can be used to perform tasks.
- Grounds system’s knowledge to prevent hallucinations

Overview of CoScientist



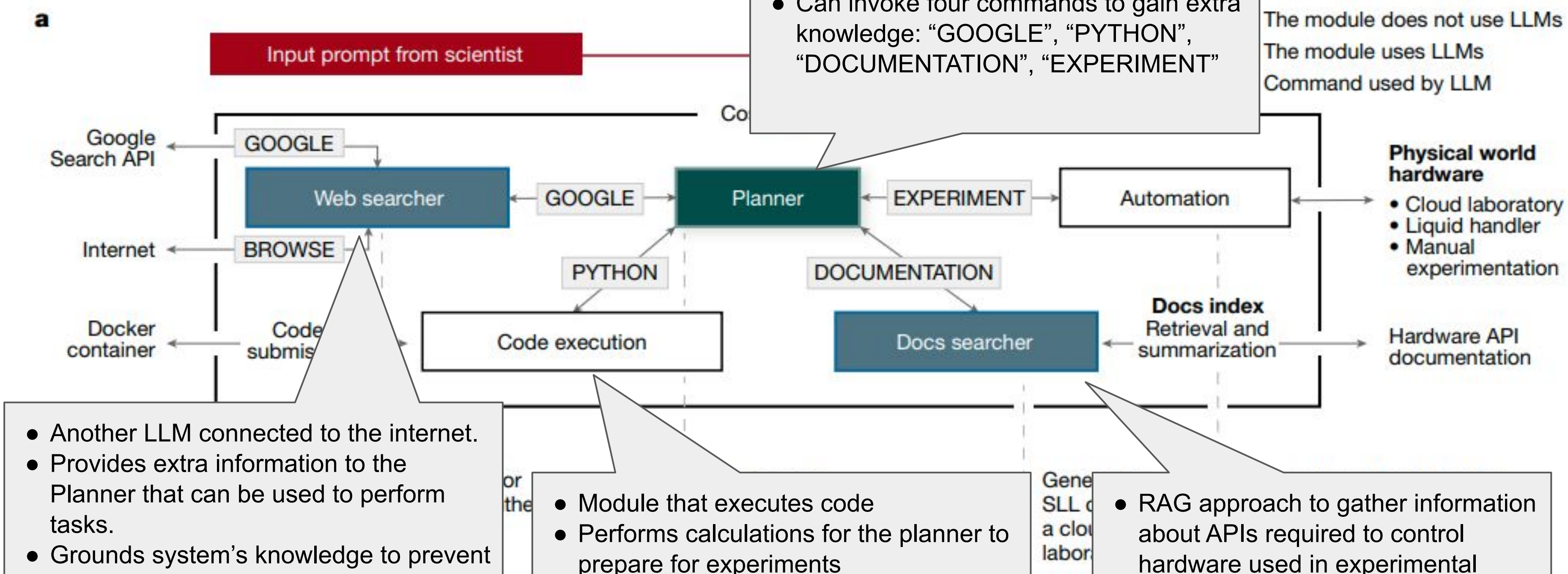
- Pretained LLM like GPT-4.
- Responsible for coordinating tasks to complete task provided by input prompt
- Can invoke four commands to gain extra knowledge: “GOOGLE”, “PYTHON”, “DOCUMENTATION”, “EXPERIMENT”

- Another LLM connected to the internet.
- Provides extra information to the Planner that can be used to perform tasks.
- Grounds system’s knowledge to prevent hallucinations

- Module that executes code
- Performs calculations for the planner to prepare for experiments
- Performs code execution in Docker container to protect user’s machine

Overview of CoScientist

- Pretained LLM like GPT-4.
- Responsible for coordinating tasks to complete task provided by input prompt
- Can invoke four commands to gain extra knowledge: “GOOGLE”, “PYTHON”, “DOCUMENTATION”, “EXPERIMENT”

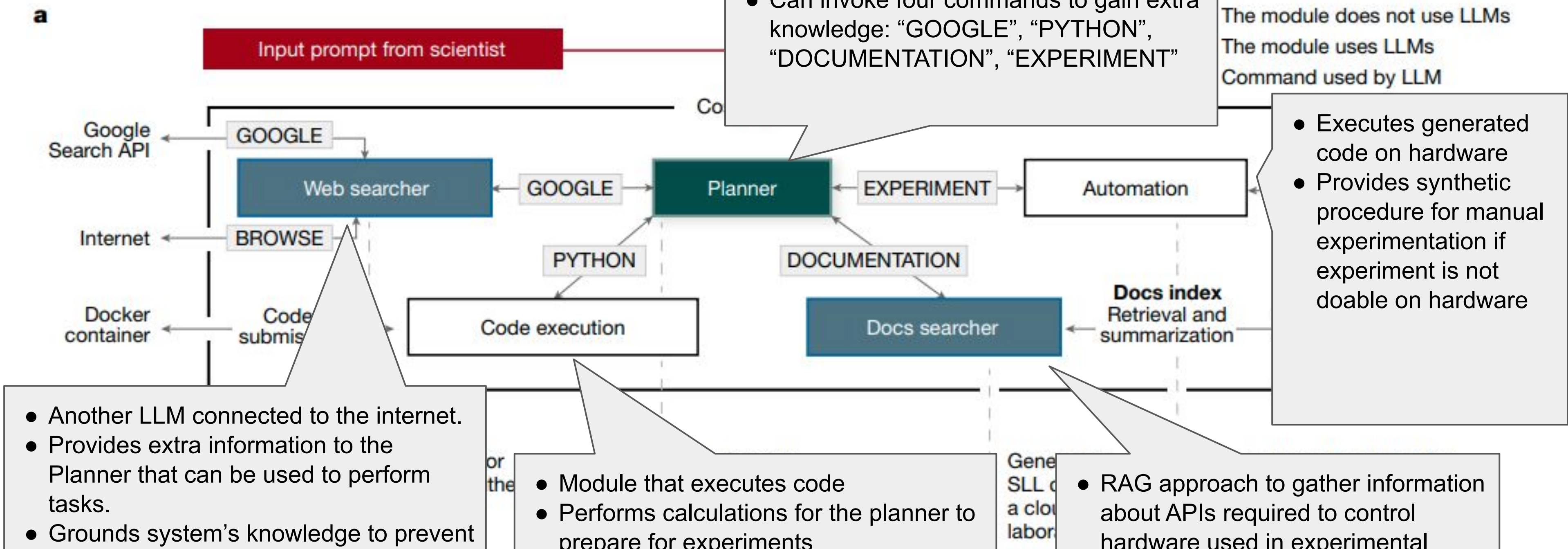


- Another LLM connected to the internet.
- Provides extra information to the Planner that can be used to perform tasks.
- Grounds system’s knowledge to prevent hallucinations

- Module that executes code
- Performs calculations for the planner to prepare for experiments
- Performs code execution in Docker container to protect user’s machine

- RAG approach to gather information about APIs required to control hardware used in experimental setups.

Overview of CoScientist



- Pretained LLM like GPT-4.
- Responsible for coordinating tasks to complete task provided by input prompt
- Can invoke four commands to gain extra knowledge: “GOOGLE”, “PYTHON”, “DOCUMENTATION”, “EXPERIMENT”

- Executes generated code on hardware
- Provides synthetic procedure for manual experimentation if experiment is not doable on hardware

- Another LLM connected to the internet.
- Provides extra information to the Planner that can be used to perform tasks.
- Grounds system’s knowledge to prevent hallucinations

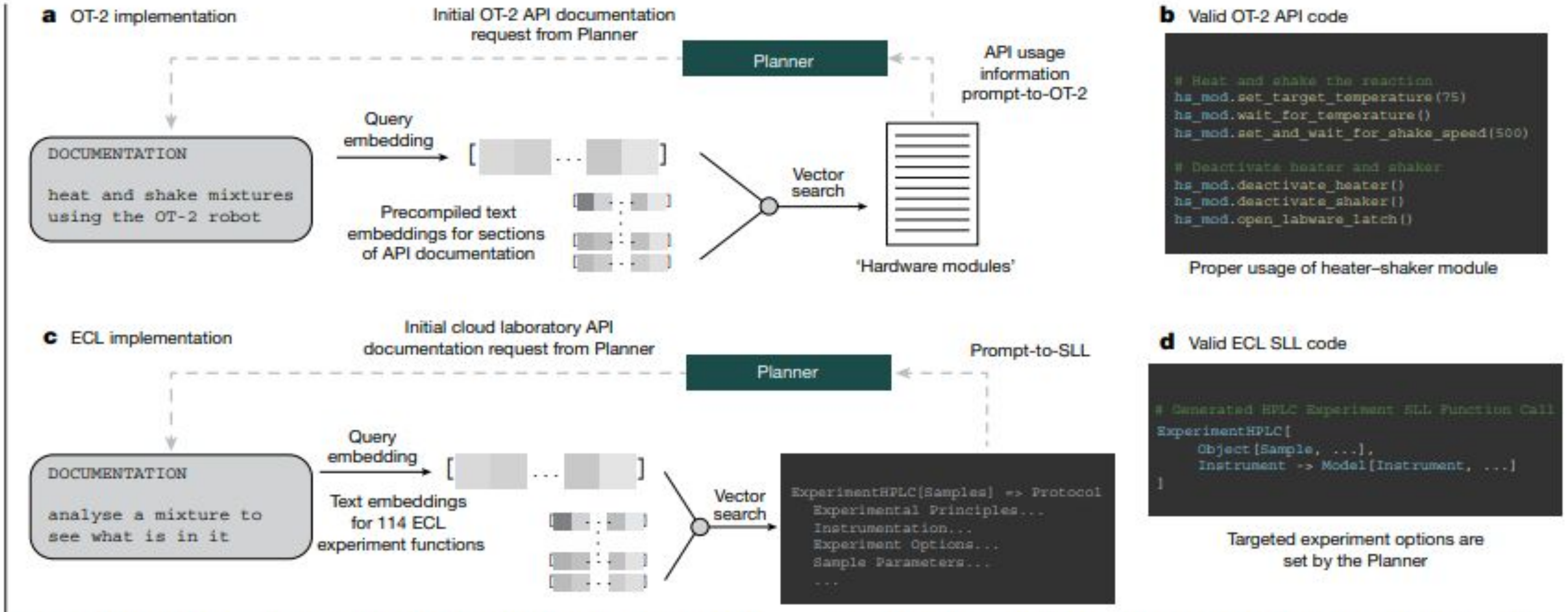
- Module that executes code
- Performs calculations for the planner to prepare for experiments
- Performs code execution in Docker container to protect user’s machine

- RAG approach to gather information about APIs required to control hardware used in experimental setups.
- Utilizes another separate LLM external to the Planner

Documentation Search Module Implementation Details

- Used a vector database approach to search for relevant documentation.
- Steps for search for relevant documentation using vector database approach:
  1. Chunk your documentation into appropriate sizes
  2. Use a model like OpenAI’s ada to convert each chunk into an embeddings
  3. To query, convert the query into an embedding
  4. Find relevant documentation chunks by calculating the similarity between the query embedding and chunk embedding using a distance based metric to assess similarity.

Documentation Search Module Implementation Details


a OT-2 implementation
Initial OT-2 API documentation request from Planner
Planner
API usage information prompt-to-OT-2
DOCUMENTATION
heat and shake mixtures using the OT-2 robot
Query embedding
Precompiled text embeddings for sections of API documentation
Vector search
'Hardware modules'
b Valid OT-2 API code
Proper usage of heater–shaker module
c ECL implementation
Initial cloud laboratory API documentation request from Planner
Planner
Prompt-to-SLL
DOCUMENTATION
analyse a mixture to see what is in it
Query embedding
Text embeddings for 114 ECL experiment functions
Vector search
d Valid ECL SLL code
Targeted experiment options are set by the Planner

```
# Heat and shake the reaction
hs_mod.set_target_temperature(75)
hs_mod.wait_for_temperature()
hs_mod.set_and_wait_for_shake_speed(500)

# Deactivate heater and shaker
hs_mod.deactivate_heater()
hs_mod.deactivate_shaker()
hs_mod.open_labware_latch()
```

```
ExperimentHPLC[Samples] => Protocol
  Experimental Principles...
  Instrumentation...
  Experiment Options...
  Sample Parameters...
  ...
```

```
# Generated HPLC Experiment SLL Function Call
ExperimentHPLC[
     Object[Sample, ...],
     Instrument -> Model[Instrument, ...]
]
```

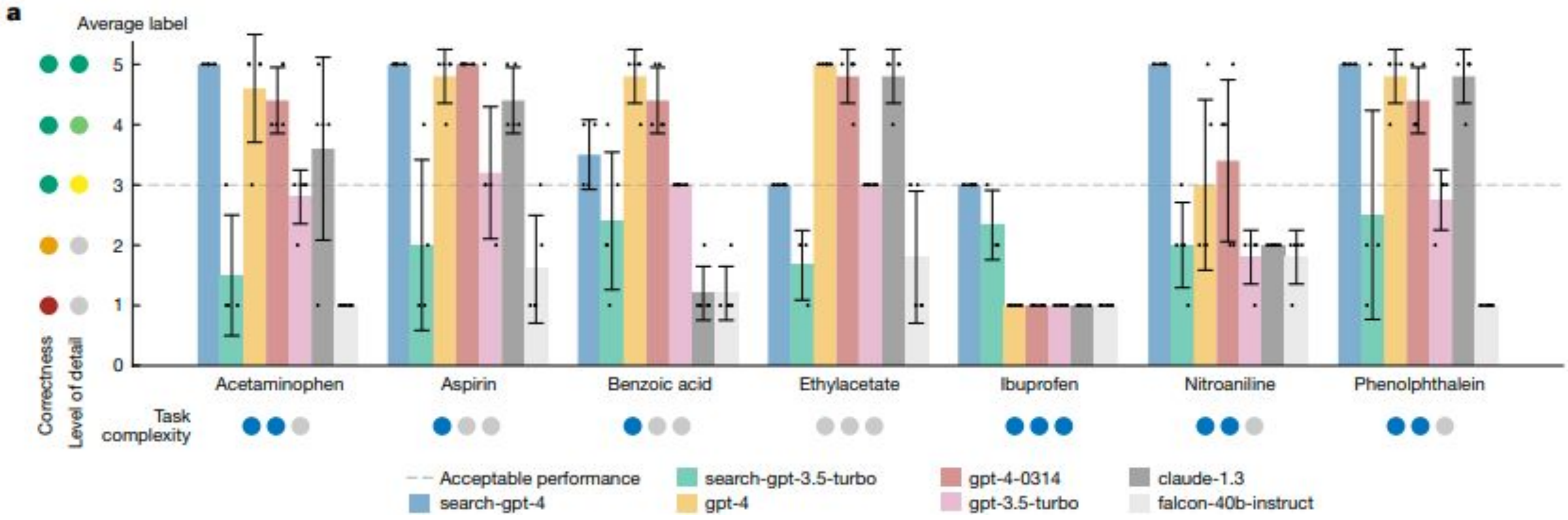
Fig. 3 | Overview of documentation search. a, Prompt-to-code through ada embedding and distance-based vector search. **b**, Example of code for using OT-2's heater–shaker module. **c**, Prompt-to-function/prompt-to-SLL (to symbolic laboratory language) through supplementation of documentation. **d**, Example of valid ECL SLL code for performing high-performance liquid chromatography (HPLC) experiments.

Experiments

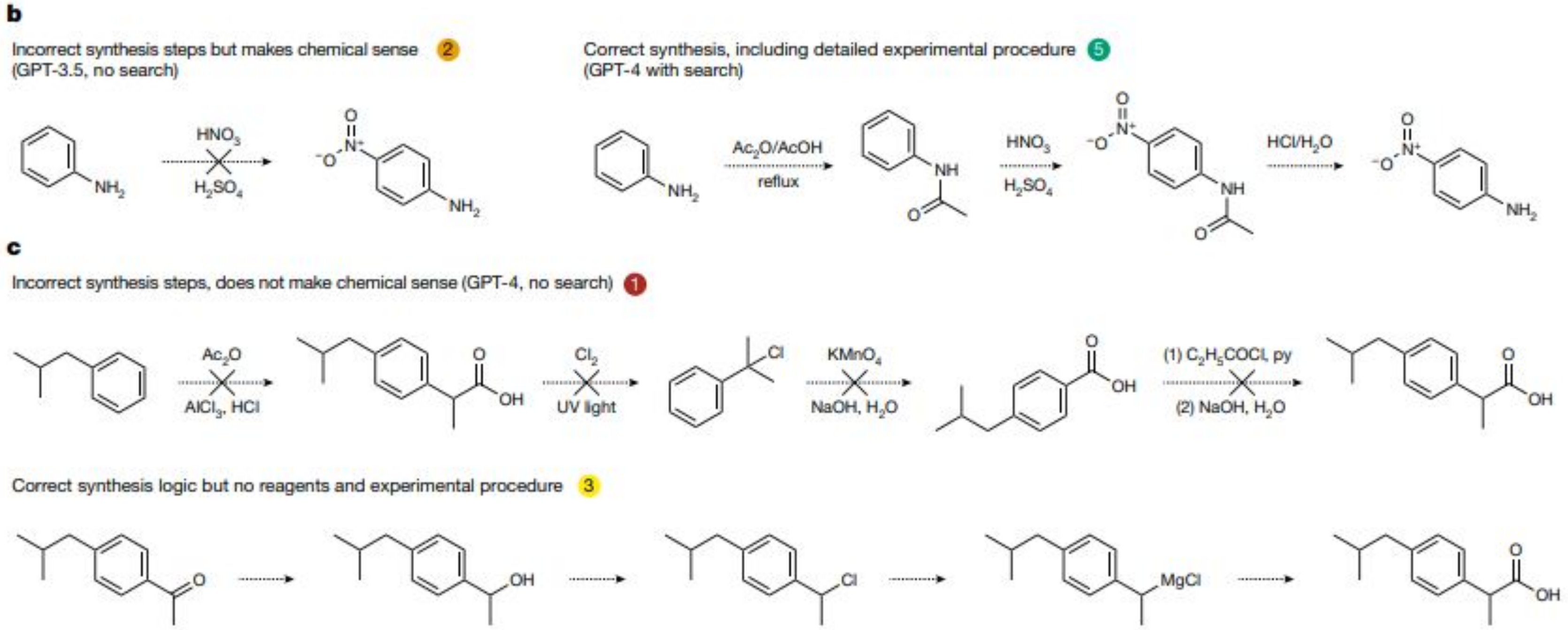
Assessing Web Search Module Capabilities

- Multiple language models were assessed to see how well they can serve as a web searcher. (search-gpt-4, search-gpt-3.5-turbo, GPT-3.5, GPT-4, Claude 1.3, Falcon-40B-Instruct)
- Test set was designed in which models were tasked with synthesizing seven compounds.
- Models scored between 1-5 in their ability to provide a detailed compound synthesis.

Assessing Web Search Module Capabilities



Assessing Web Search Module Capabilities



b

Incorrect synthesis steps but makes chemical sense 2 (GPT-3.5, no search)

Correct synthesis, including detailed experimental procedure 5 (GPT-4 with search)

c

Incorrect synthesis steps, does not make chemical sense (GPT-4, no search) 1

Correct synthesis logic but no reagents and experimental procedure 3

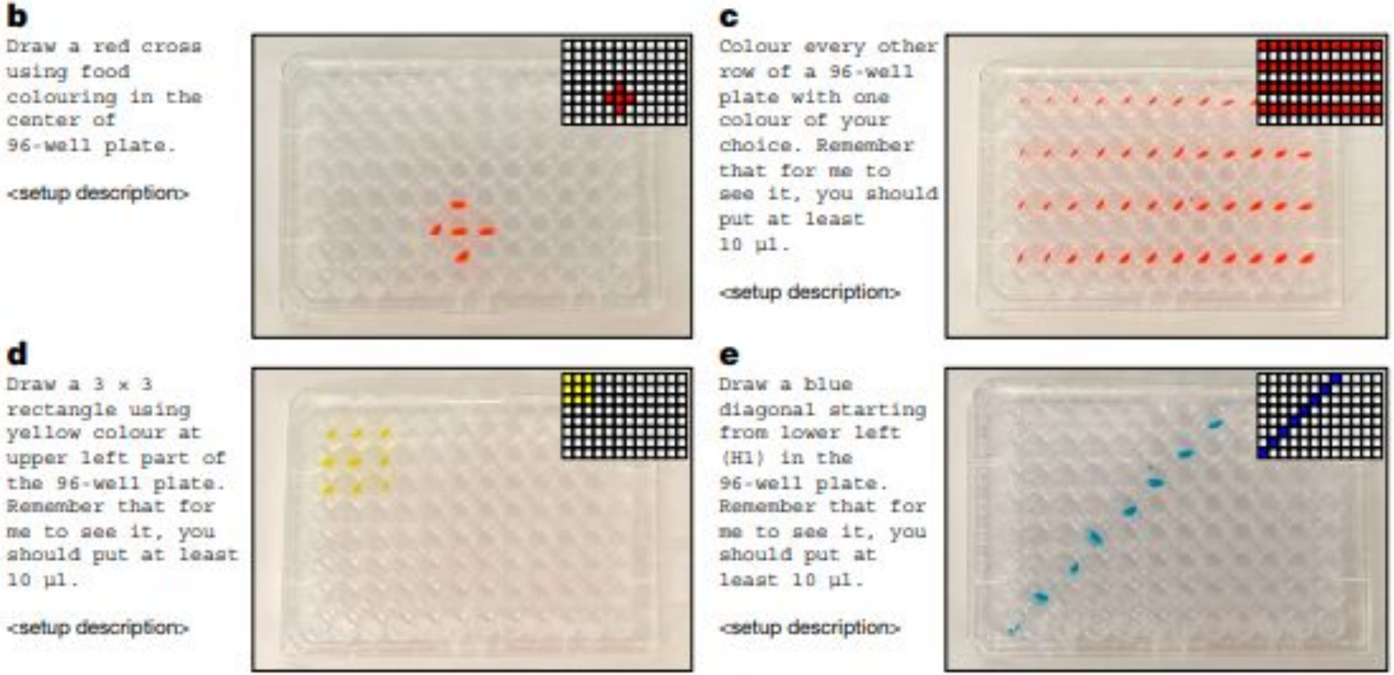
Fig. 2 | Coscientist's capabilities in chemical synthesis planning tasks. a, Comparison of various LLMs on compound synthesis benchmarks. Error bars represent s.d. values. **b**, Two examples of generated syntheses of nitroaniline. **c**, Two example of generated syntheses of ibuprofen. UV, ultraviolet.

Abilities in Controlling Laboratory Hardware

- The system's ability was assessed through multiple experiments:
    - Simple plate-layout specific experiments like "color every other line with one color of your choice". Just requires control of a liquid handler to solve problem.

    - Tasks that involve the integration of multiple modules. For example, "In 3 wells of a 96-well plate, three different colours are present—red, yellow and blue. Determine the colors and their positions on the plate. This task requires both the liquid handler and UV-Vis plate reader to solve the problem.

Abilities in Controlling Laboratory Hardware



b

Draw a red cross using food colouring in the center of 96-well plate.

<setup description>

c

Colour every other row of a 96-well plate with one colour of your choice. Remember that for me to see it, you should put at least 10 µl.

<setup description>

d

Draw a 3 x 3 rectangle using yellow colour at upper left part of the 96-well plate. Remember that for me to see it, you should put at least 10 µl.

<setup description>

e

Draw a blue diagonal starting from lower left (H1) in the 96-well plate. Remember that for me to see it, you should put at least 10 µl.

<setup description>

Integrated Chemical Experiment Design

- Test to see how well CoScientist can use all of its tools to perform catalytic cross-coupling experiments.
- Performs the experiment in the following steps:
    1. Uses internet to find appropriate reaction conditions and stoichiometries
    2. Calculates required reagent volumes and generates hardware protocols
    3. Successfully identifies and corrects errors in hardware module method names
    4. Gas chromatography confirms formation of target products for both reactions
    5. Shows reasoning capabilities about reagent selection and provides justifications

Key Takeaways

- Coscientist, powered by GPT-4, autonomously plans and executes chemical experiments using tools like web search, code execution, and lab APIs.

- Has the ability to accurately perform complex tasks such as synthesis planning, hardware control, and experiment automation with minimal human input.

- Validated through real-world experiments (e.g., Suzuki and Sonogashira reactions), Coscientist showcases the potential of LLMs in accelerating scientific discovery.

Paper 2

Article

https://doi.org/10.1038/s42256-024-00832-8

Augmenting large language models with chemistry tools

Received: 13 September 2023

Accepted: 27 March 2024

Andres M. Bran[1,2,6], Sam Cox[3,4,6], Oliver Schilter[1,2,5], Carlo Baldassari[5], Andrew D. White[3,4] & Philippe Schwaller[1,2]

Paper’s Contributions

- Introduce ChemCrow, a chemistry LLM agent built on GPT-4 to accomplish tasks across organic synthesis, drug discovery, and materials design.

- Integrates 18 expert-designed tools spanning synthesis, safety, molecule analysis, and more.

- Validated effectiveness via real-world synthesis of an insect repellent, organocatalysts, and chromophore discovery.

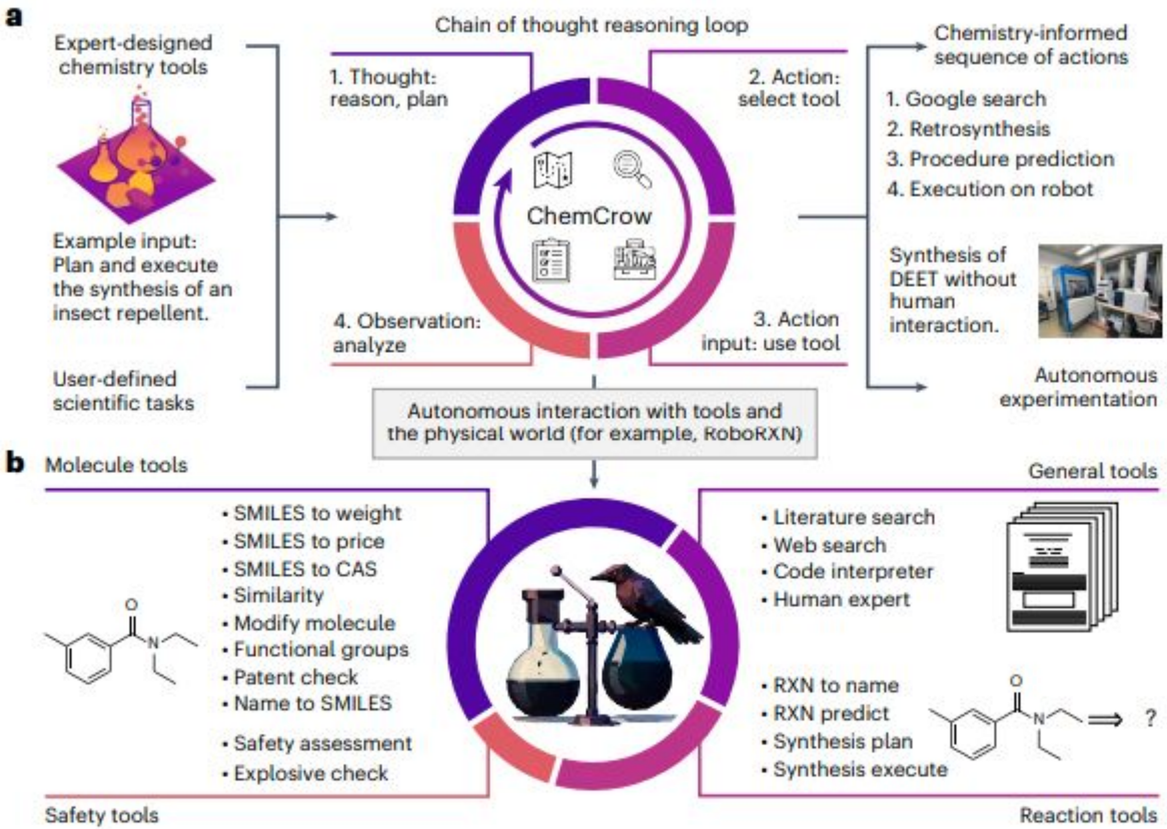
Overview of ChemCrow

- LLM provided with a list of tools names, descriptions of their utility, and details of expected input/output.

- Model is guided to follow Thought, Action, Action Input, and Observation format to allow for chain-of-thought reasoning.

- By augmenting the model with tools, the aim is to transition the model from a hyper-confident hallucinating model to a reasoning engine that reflects on a task and uses tools to gather information.

Overview of ChemCrow

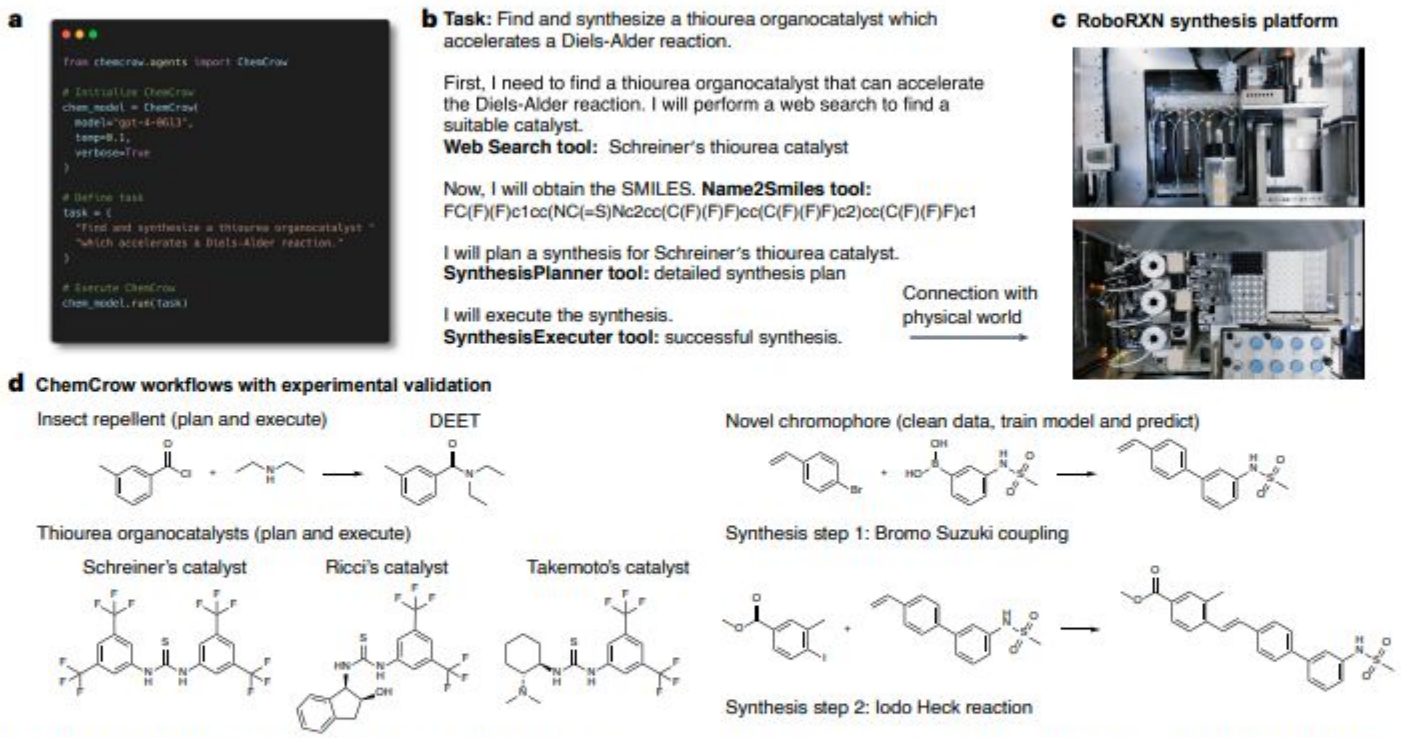


Tools Used for LLM Augmentation

- General Tools: Web search, Python REPL, LitSearch
- Molecule Tools: Name2SMILES, SMILES2Price, PatentCheck, SMILES2Weight.
- Reaction Tools: ReactionPredict, SynthesisPlanner, ReactionExecute.
- Safety Tools: ControlledChemicalCheck, ExplosiveCheck, SafetySummary.

Experiments and Results

Autonomous Chemical Synthesis



a

```
from chemcrow.agents import ChemCrow

# Initialize ChemCrow
chem_model = ChemCrow(
  model="gpt-4-0613",
  temp=0.1,
  verbose=True
)

# Define task
task = (
  "Find and synthesize a thiourea organocatalyst "
  "which accelerates a Diels-Alder reaction."
)

# Execute ChemCrow
chem_model.run(task)
```

b **Task:** Find and synthesize a thiourea organocatalyst which accelerates a Diels-Alder reaction.

First, I need to find a thiourea organocatalyst that can accelerate the Diels-Alder reaction. I will perform a web search to find a suitable catalyst.
Web Search tool: Schreiner's thiourea catalyst

Now, I will obtain the SMILES. **Name2Smiles tool:**
FC(F)(F)c1cc(NC(=S)Nc2cc(C(F)(F)F)cc(C(F)(F)F)c2)cc(C(F)(F)F)c1

I will plan a synthesis for Schreiner's thiourea catalyst.
SynthesisPlanner tool: detailed synthesis plan

I will execute the synthesis.
SynthesisExecuter tool: successful synthesis.

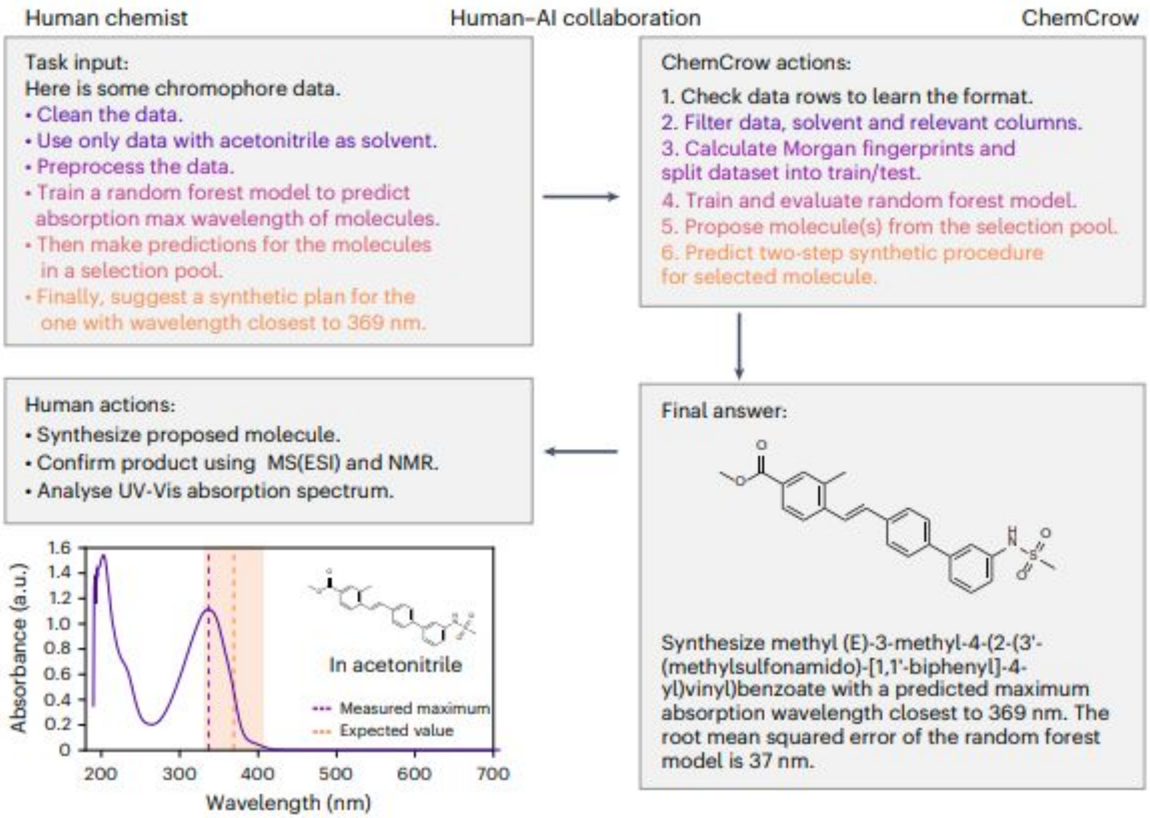
c **RoboRXN synthesis platform**

d **ChemCrow workflows with experimental validation**

Connection with physical world
Insect repellent (plan and execute)
DEET
Thiourea organocatalysts (plan and execute)
Schreiner's catalyst
Ricci's catalyst
Takemoto's catalyst
Novel chromophore (clean data, train model and predict)
Synthesis step 1: Bromo Suzuki coupling
Synthesis step 2: Iodo Heck reaction

Fig. 2 | Experimental validation. **a**, Example of the script run by a user to initiate ChemCrow. **b**, Query and synthesis of a thiourea organocatalyst. **c**, IBM Research RoboRXN synthesis platform on which the experiments were executed (pictures reprinted courtesy of International Business Machines Corporation). **d**, Experimentally validated compounds. Credit: photographs in **c**, IBM Research under a creative commons license CC BY-ND 2.0.

Human-AI Collaboration



Human chemist

Human-AI collaboration

ChemCrow

Task input:
Here is some chromophore data.

- Clean the data.
- Use only data with acetonitrile as solvent.
- Preprocess the data.
- Train a random forest model to predict absorption max wavelength of molecules.
- Then make predictions for the molecules in a selection pool.
- Finally, suggest a synthetic plan for the one with wavelength closest to 369 nm.

ChemCrow actions:

1. Check data rows to learn the format.
2. Filter data, solvent and relevant columns.
3. Calculate Morgan fingerprints and split dataset into train/test.
4. Train and evaluate random forest model.
5. Propose molecule(s) from the selection pool.
6. Predict two-step synthetic procedure for selected molecule.

Final answer:

Synthesize methyl (E)-3-methyl-4-(2-(3'-(methylsulfonamido)-[1,1'-biphenyl]-4-yl)vinyl)benzoate with a predicted maximum absorption wavelength closest to 369 nm. The root mean squared error of the random forest model is 37 nm.

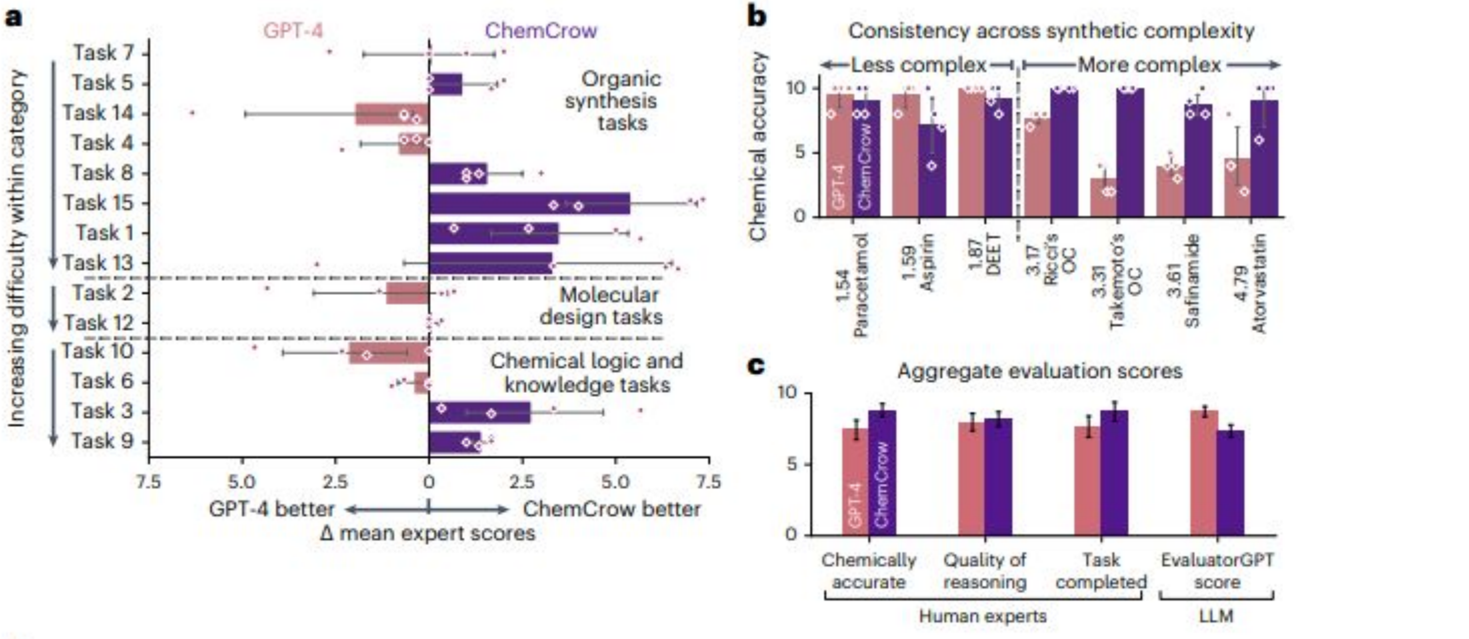
Human actions:

- Synthesize proposed molecule.
- Confirm product using MS(ESI) and NMR.
- Analyse UV-Vis absorption spectrum.

Evaluation Across Diverse Chemical Use Cases

- Compared ChemCrow vs. GPT-4 across 14 tasks using expert and LLM assessments.
- ChemCrow outperformed GPT-4 in chemical accuracy and task completion.
- GPT-4 responses were more fluent but often incorrect.
- Human experts preferred ChemCrow; LLM evaluators showed GPT-4 bias.

Evaluation Across Diverse Chemical Use Cases



GPT-4 (without tools)

- ✓ Complete responses (when possible)
- ☒ Major hallucinations (molecules, reactions, procedures)
- ☒ Hard to interpret (need for expert modifications)
- ☒ No access to up-to-date information

ChemCrow

- ✓ Chemically accurate solutions
- ✓ Modular and extensible
- ☒ Occasional flawed conclusions
- ☒ Limited by tools' quality

Key Takeaways

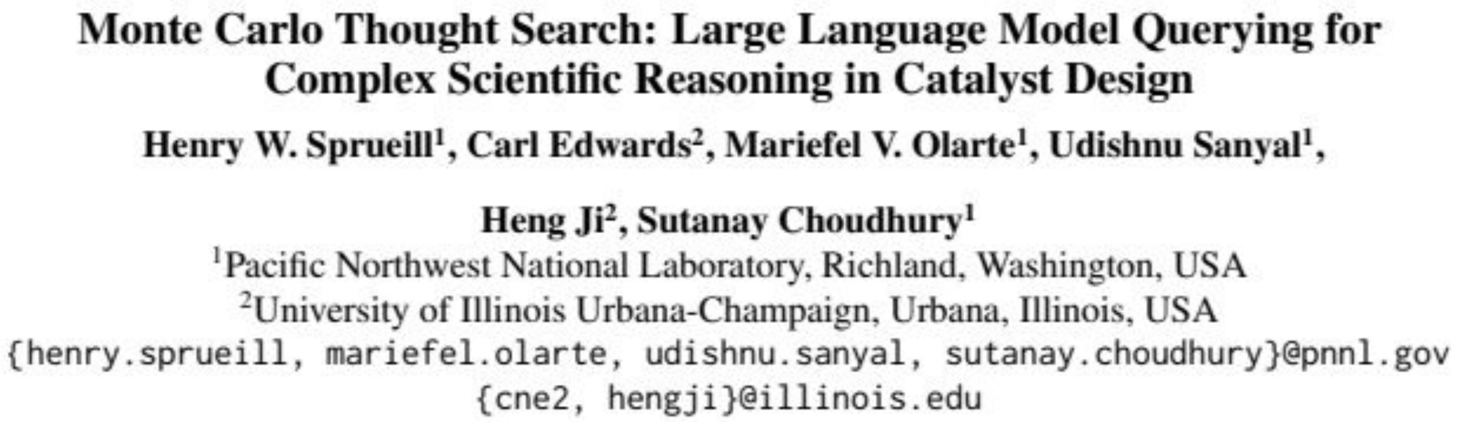
- **ChemCrow transforms LLMs into practical chemistry agents** by integrating 18 domain-specific tools that handle tasks like reaction prediction, molecule modification, safety checks, and synthesis execution, going far beyond the capabilities of standalone LLMs.

- **The agent can autonomously perform complex experimental tasks** such as synthesizing insect repellents and organocatalysts, by planning multistep syntheses, adapting to errors in robotic platforms, and executing physical experiments with no human intervention.

- **ChemCrow supports meaningful human-AI collaboration**, demonstrated through successful joint efforts like machine learning-driven chromophore discovery—where the model cleaned data, trained a model, and proposed synthesis, while the chemist validated the results.

- **Evaluation by expert chemists shows ChemCrow outperforms GPT-4** on chemical accuracy, reasoning, and task completion, especially on complex or novel problems that require real-world understanding and precise tool use.

Paper 3



Monte Carlo Thought Search: Large Language Model Querying for Complex Scientific Reasoning in Catalyst Design

Henry W. Sprueill[1], Carl Edwards[2], Mariefel V. Olarte[1], Udishnu Sanyal[1],

Heng Ji[2], Sutanay Choudhury[1]

[1]Pacific Northwest National Laboratory, Richland, Washington, USA
[2]University of Illinois Urbana-Champaign, Urbana, Illinois, USA
`{henry.sprueill, mariefel.olarte, udishnu.sanyal, sutanay.choudhury}@pnnl.gov`
`{cne2, hengji}@illinois.edu`

Paper’s Contributions

- Introduces the Monte Carlo Reasoner (MCR)—a method that augments large language models with tree-based reasoning for scientific tasks like catalyst discovery.
- Develops a new chemistry-focused dataset—BioFuelQR—to evaluate LLM reasoning on realistic chemical problems faced daily by scientists.
- Incorporates a domain-specific reward function (based on adsorption energy) to assess the quality of LLM-generated catalysts.
- Achieves significant improvements over existing prompting techniques like Chain-of-Thought (CoT) and Tree-of-Thought (ToT),

Problems With LLM Querying Currently

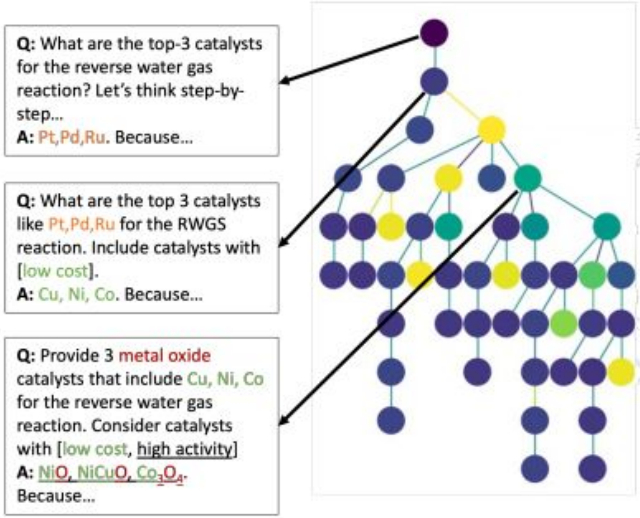
- Prompts like “What is a good catalyst for reaction X?” yield answers no better than a search engine; they lack scientific depth and justification.

- Answers generated lack domain specificity and key technical terminology.

- There's a high risk of hallucinations, where the model fabricates confident but incorrect information, undermining trust in its outputs.

Monte-Carlo Reasoner: Overview and Goal

- MCR is a zero-shot prompting strategy that guides LLMs to perform structured reasoning for scientific tasks without additional model training.
- Uses Monte Carlo Tree Search (MCTS) to explore and expand prompts by applying domain-inspired actions (e.g., adding constraints like low toxicity or catalyst type)
- Each prompt P is a node in the tree.
- Each action a∈A modifies the prompt (e.g., add a constraint).
- Each edge (P,a) leads to a new prompt.
- The LLM's response to a prompt gives a reward R(P), which scores the quality of the suggested catalysts.
- The search proceeds by simulating different prompt sequences and updating estimates based on outcomes.
- Goal is to find prompt P°, which is the optimal prompt.



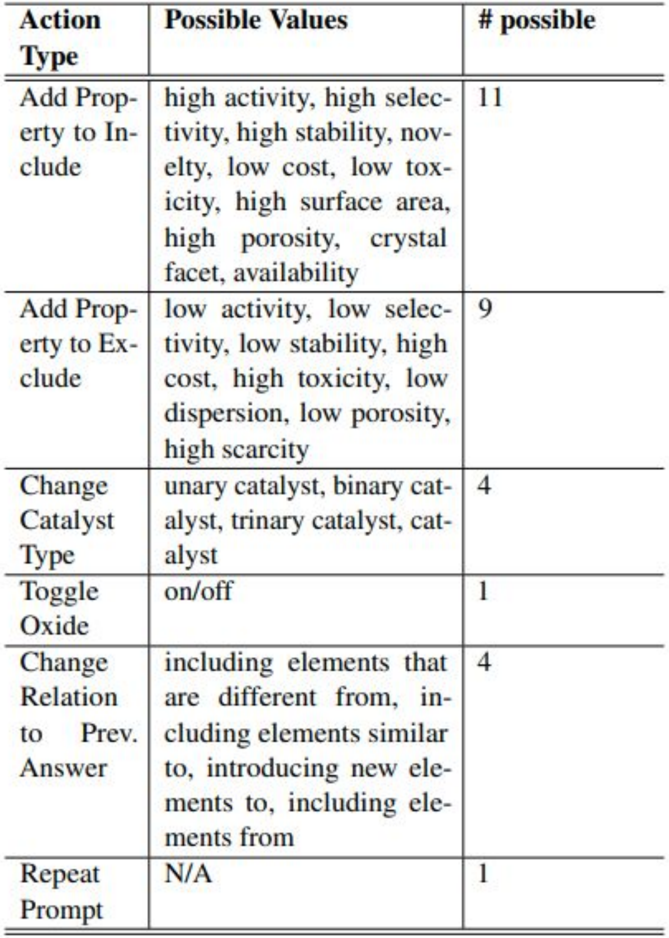
Q: What are the top-3 catalysts for the reverse water gas reaction? Let's think step-by-step...
A: Pt,Pd,Ru. Because...

Q: What are the top 3 catalysts like Pt,Pd,Ru for the RWGS reaction. Include catalysts with [low cost].
A: Cu, Ni, Co. Because...

Q: Provide 3 metal oxide catalysts that include Cu, Ni, Co for the reverse water gas reaction. Consider catalysts with [low cost, high activity]
A: NiO, NiCuO, Co_3O_4. Because...

Monte-Carlo Reasoner: Actions

Table 3: List of actions and their possibilities.



Action Type	Possible Values	# possible
Add Property to Include	high activity, high selectivity, high stability, novelty, low cost, low toxicity, high surface area, high porosity, crystal facet, availability	11
Add Property to Exclude	low activity, low selectivity, low stability, high cost, high toxicity, low dispersion, low porosity, high scarcity	9
Change Catalyst Type	unary catalyst, binary catalyst, trinary catalyst, catalyst	4
Toggle Oxide	on/off	1
Change Relation to Prev. Answer	including elements that are different from, including elements similar to, introducing new elements to, including elements from	4
Repeat Prompt	N/A	1

Monte-Carlo Reasoner: Steps

1. Initialize the search tree
2. Simulate the tree search
    2.1. Action Selection
    2.2. Tree Expansion
    2.3. Reward calculation
3. Find prompt with maximum reward

Monte-Carlo Reasoner: 1. Initialize the Search Tree

- We start of with providing our start prompt P_0. This prompt must be of the form of “What are the top-k catalysts for…”

- This prompt will serve as the root of the tree root(T).

Monte-Carlo Reasoner: 2. Simulate the tree search

- We now start the tree search loop. In each iteration of the loop we:
    a. Traverse the tree using action selection

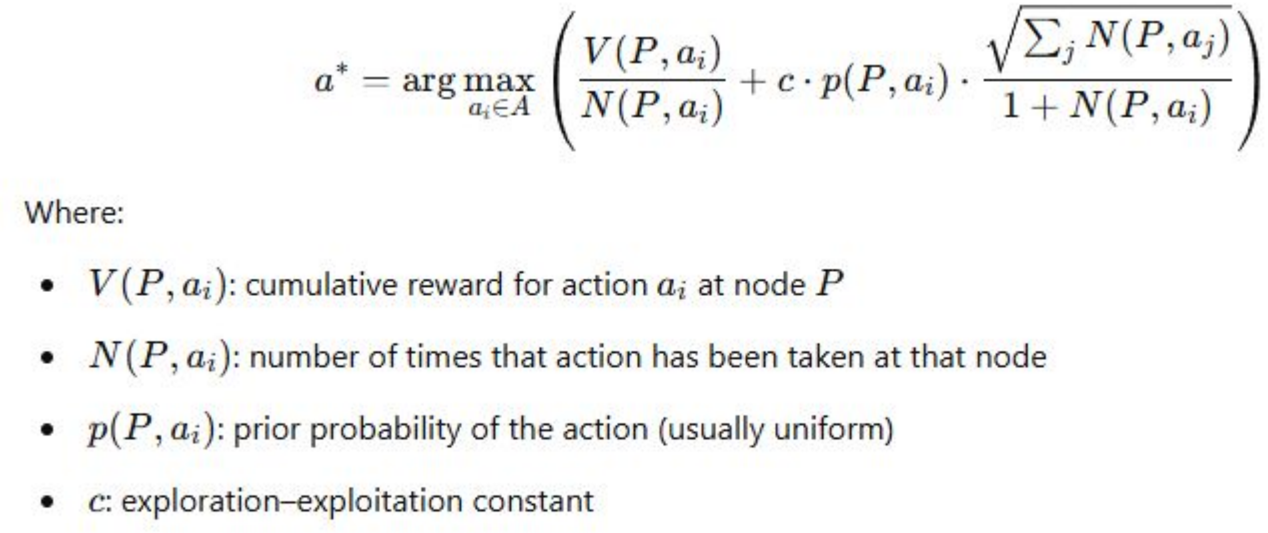
    b. Expand the tree by adding a new leaf (prompt P_t)

    c. Calculate the reward of the prompt P_t and update downstream rewards V(P, a).

- We repeat this tree search M times, where M is the number of candidate prompts we want to generate.

Monte-Carlo Reasoner: 2.1. Traverse the Tree

- During each tree traversal we start at the root of the tree.
- We decide which child node to go to using this UCB-like formula:



$$a^* = \arg\max_{a_i \in A} \left(\frac{V(P, a_i)}{N(P, a_i)} + c \cdot p(P, a_i) \cdot \frac{\sqrt{\sum_j N(P, a_j)}}{1 + N(P, a_i)} \right)$$

Where:

- $V(P, a_i)$: cumulative reward for action a_i at node P
- $N(P, a_i)$: number of times that action has been taken at that node
- $p(P, a_i)$: prior probability of the action (usually uniform)
- c: exploration–exploitation constant

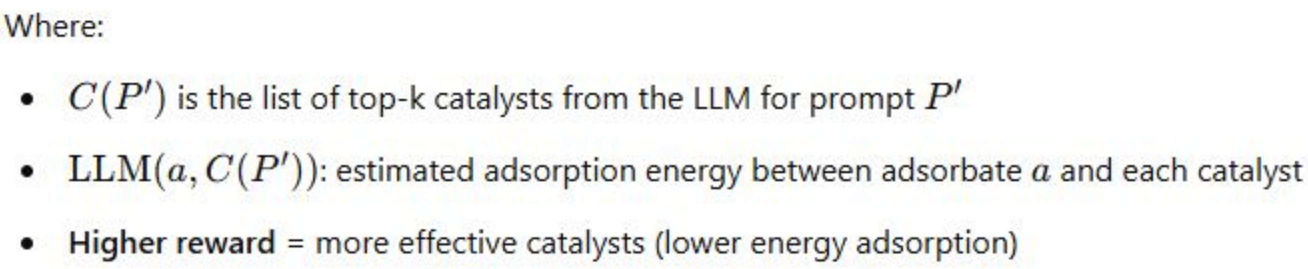
Monte-Carlo Reasoner: 2.2. Expand the Tree

- When we reach a leaf node after tree traversal, we expand the tree by adding a new prompt P’ = $a^t(P_t)$. Here a^t is calculated by the UCB formula at the leaf step.

Monte-Carlo Reasoner: 2.3. Reward Calculation

- Calculate reward for prompt P’ as:

$$R(P') = \sum_{a \in \text{adsorbates}} |\text{LLM}(a, C(P'))|$$



Where:

- $C(P')$ is the list of top-k catalysts from the LLM for prompt P'
- $\text{LLM}(a, C(P'))$: estimated adsorption energy between adsorbate a and each catalyst
- **Higher reward** = more effective catalysts (lower energy adsorption)

- Update all the cumulative rewards for all *t* using new reward:

$$V(P_t, a_t) \leftarrow V(P_t, a_t) + \gamma^d R(P')$$

Monte-Carlo Reasoner: 3. Find prompt With Maximum Reward

- Traverse through the tree after M steps and find node (prompt) with maximum reward:

$$P^o = \arg\max_{P \in T} R(P)$$

Experiments

Baselines and Datasets

- Baselines:
  - **Chain of Thought (CoT):** A reasoning approach where an LLM generates step-by-step intermediate thoughts to arrive at a final answer.
  - **Self-Consistency CoT:** An extension of CoT where multiple reasoning paths are sampled and the most consistent answer is selected via majority voting.
  - **Tree of Thought (ToT):** A structured reasoning framework where an LLM explores multiple reasoning branches in a tree-like manner, evaluating and expanding promising paths.
- Datasets: OpenCatalysis (adsorption focus) and BioFuelQR (biofuels reasoning).

Results

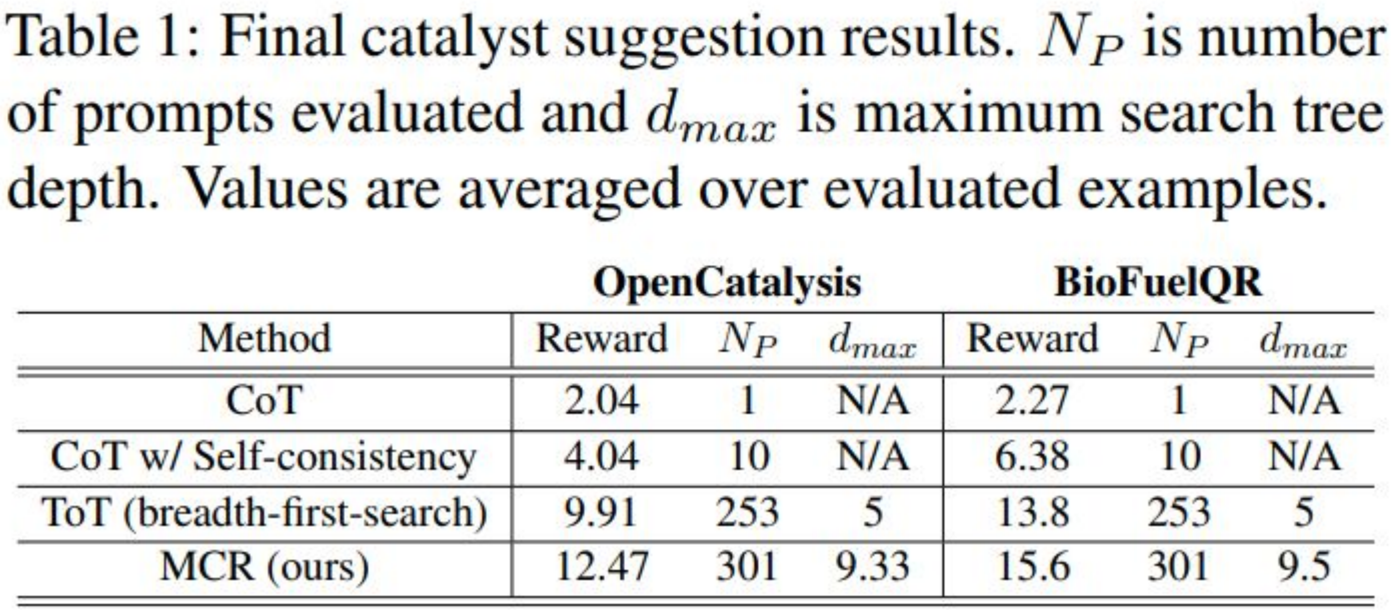


Table 1: Final catalyst suggestion results. N_P is number of prompts evaluated and d_{max} is maximum search tree depth. Values are averaged over evaluated examples.

	OpenCatalysis			**BioFuelQR**		
Method	Reward	N_P	d_{max}	Reward	N_P	d_{max}
CoT	2.04	1	N/A	2.27	1	N/A
CoT w/ Self-consistency	4.04	10	N/A	6.38	10	N/A
ToT (breadth-first-search)	9.91	253	5	13.8	253	5
MCR (ours)	12.47	301	9.33	15.6	301	9.5

Key Takeaways

- The paper introduces Monte Carlo Reasoner (MCR), a method that uses Monte Carlo Tree Search (MCTS) to guide large language models (LLMs) in performing complex scientific reasoning in catalyst design.

- MCR significantly outperforms CoT, self-consistency CoT, and Tree-of-Thought (ToT) approaches on two new benchmarks: OpenCatalysis and BioFuelQR, showing up to 25.8% improvement in catalyst selection quality.

- It incorporates a domain-specific reward function (adsorption energy estimates) to guide the search toward more scientifically accurate and specific responses.

Questions?