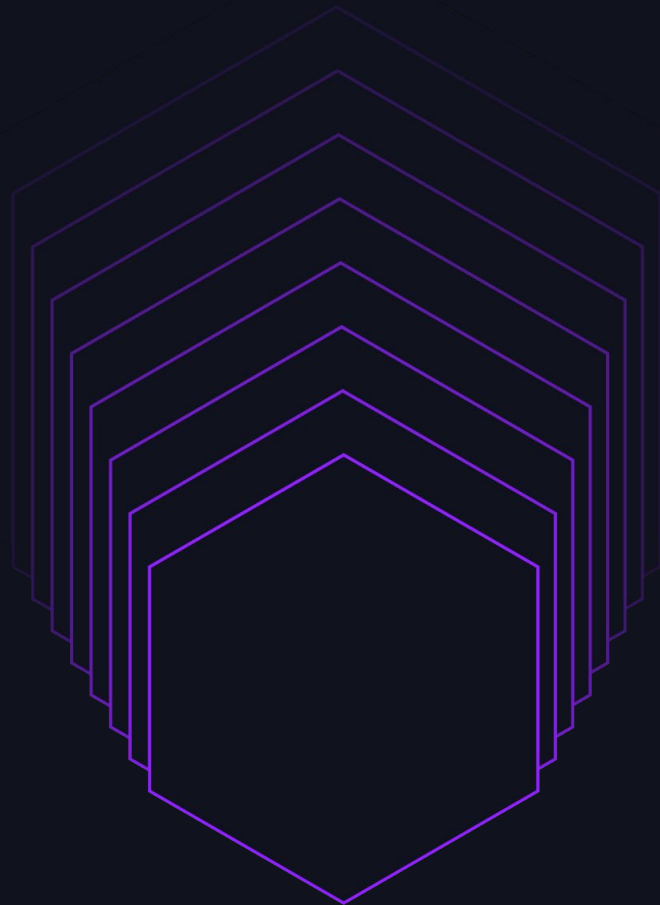


FINE-TUNING OPEN SOURCE MODELS FOR AGENTS

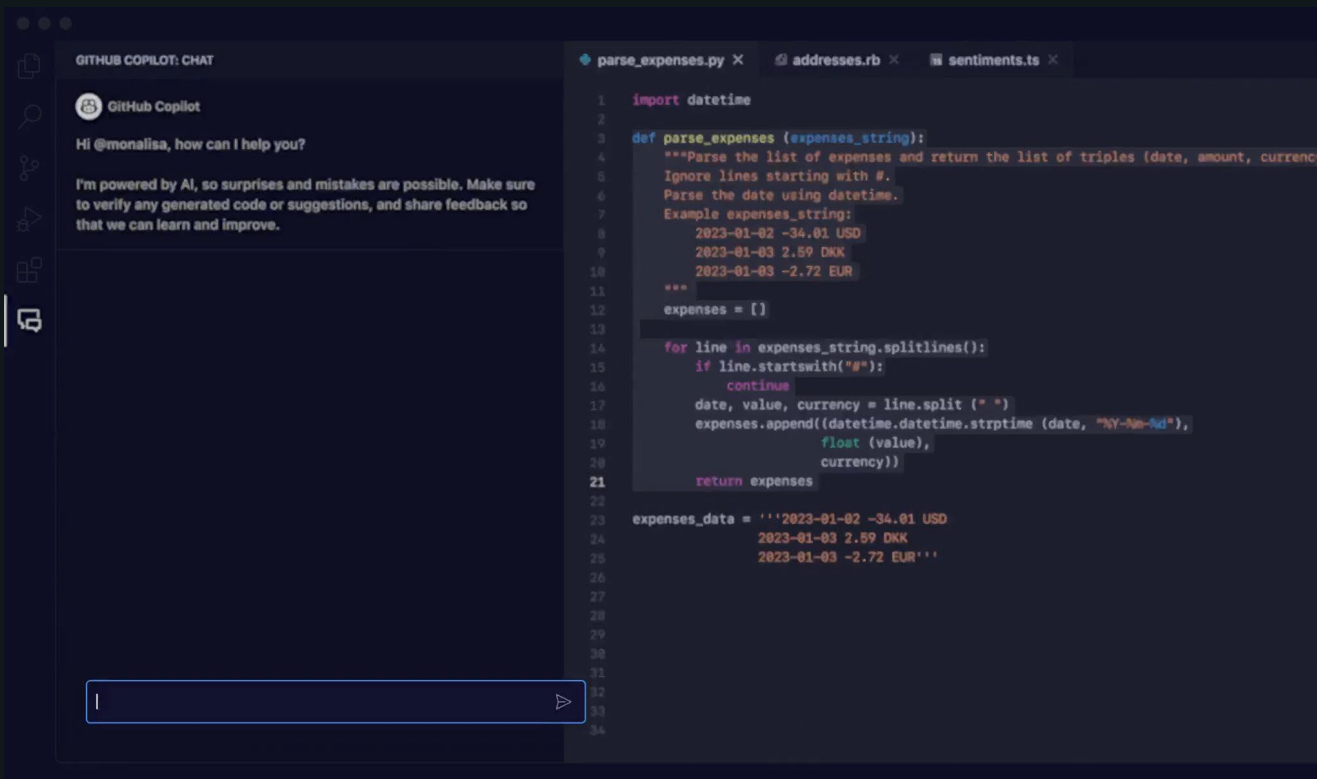
Tristan Zajonc - Continual.AI
June 11, 2024



</>

We're entering the era of AI agents

Conversational agents



The screenshot displays the GitHub Copilot chat interface. On the left, a chat window titled "GITHUB COPILOT: CHAT" shows a conversation with the GitHub Copilot agent. The user asks, "Hi @monalisa, how can I help you?" and the agent responds, "I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code or suggestions, and share feedback so that we can learn and improve." Below the chat is an input field with a cursor and a send button.

On the right, a code editor shows the file "parse_expenses.py" with the following Python code:

```
1 import datetime
2
3 def parse_expenses (expenses_string):
4     """Parse the list of expenses and return the list of triples (date, amount, currency)
5     Ignore lines starting with #.
6     Parse the date using datetime.
7     Example expenses_string:
8     2023-01-02 -34.01 USD
9     2023-01-03 2.59 DKK
10    2023-01-03 -2.72 EUR
11    """
12    expenses = []
13
14    for line in expenses_string.splitlines():
15        if line.startswith("#"):
16            continue
17        date, value, currency = line.split(" ")
18        expenses.append((datetime.datetime.strptime (date, "%Y-%m-%d"),
19                        float (value),
20                        currency))
21    return expenses
22
23 expenses_data = '''2023-01-02 -34.01 USD
24                 2023-01-03 2.59 DKK
25                 2023-01-03 -2.72 EUR'''
```

Inline agents

The screenshot shows a SQL editor window with the following elements:

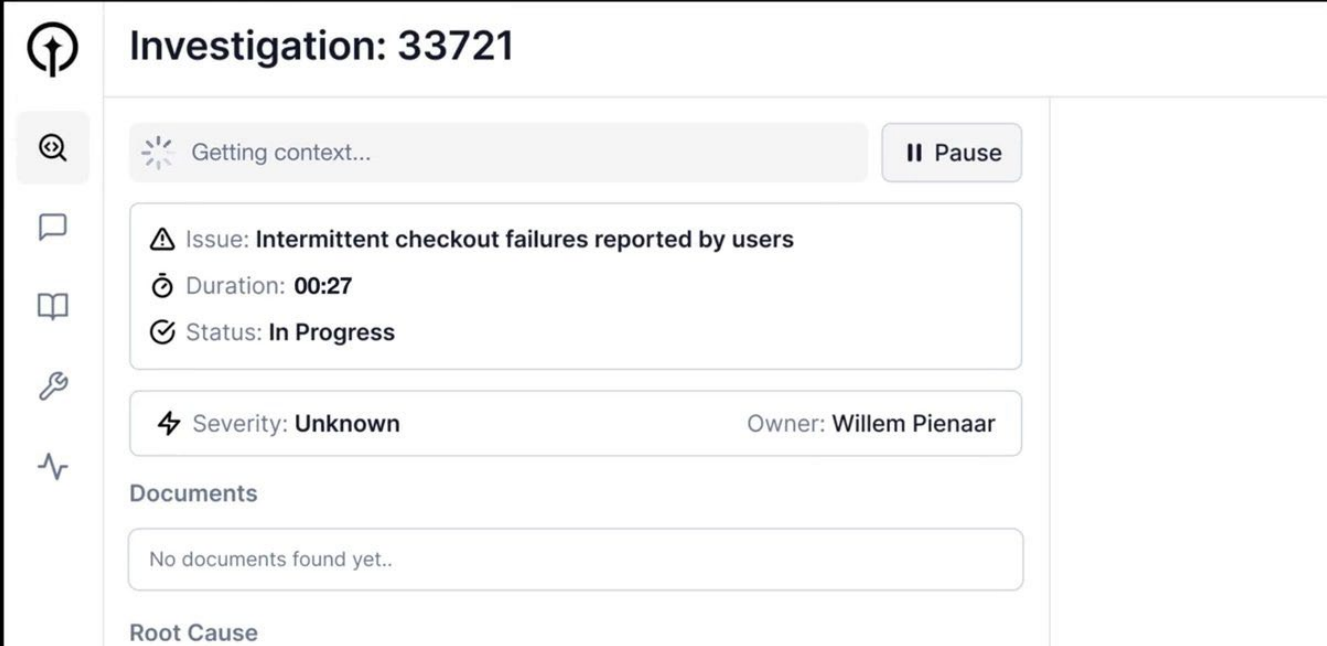
- SOURCE:** Demo Snowflake
- EDIT:** only return trips started after 6pm on weekends
- SQL Query:**

```
1 --select * from "DEMO_DATA"."DEMOS"."CITIBIKE_TRIPS"  
1+SELECT *  
2+FROM "DEMO_DATA"."DEMOS"."CITIBIKE_TRIPS"  
3+INNER JOIN "DEMO_DATA"."DEMOS"."CITIBIKE_STATIONS"  
4+ON "DEMO_DATA"."DEMOS"."CITIBIKE_TRIPS"."START_STATION_ID" = "DEMO_DATA"."DEMOS"."CITIBIKE_STATIONS"."STATION_ID"  
5+WHERE "DEMO_DATA"."DEMOS"."CITIBIKE_TRIPS"."START_TIME" > '18:00:00'  
6+AND "DEMO_DATA"."DEMOS"."CITIBIKE_STATIONS"."DAY_OF_WEEK" IN ('Saturday', 'Sunday')
```
- Buttons:** Keep (with left arrow), Reject (with ESC key)
- Warning:** Magic results may be incorrect or misleading. Learn more.
- Output:** Dataframe (with dropdown arrow)

↳ dataframe

Event-driven agents

Cleric first gathers context...



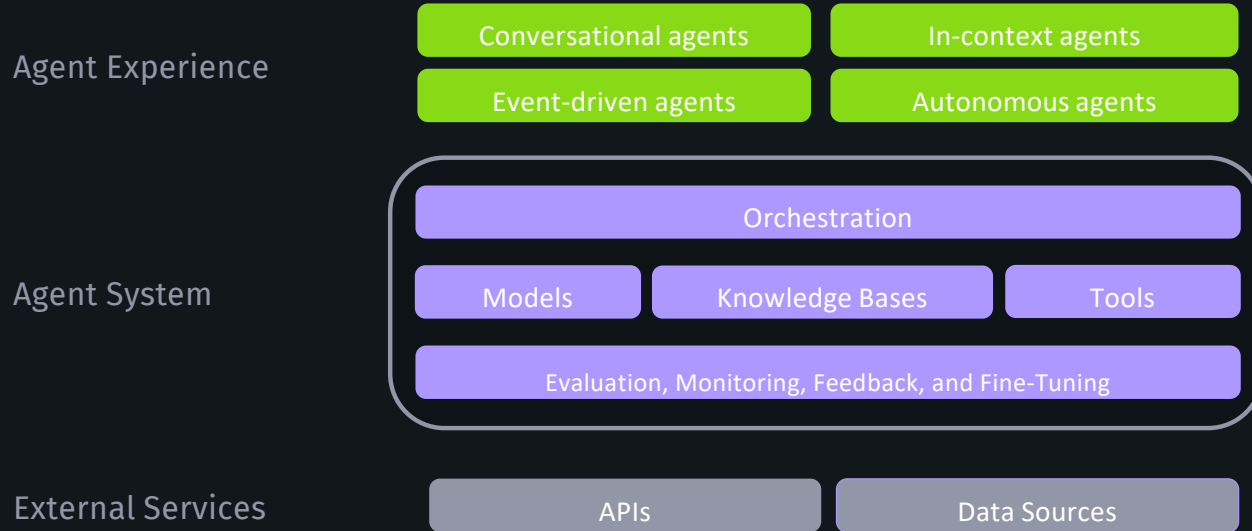
The screenshot shows the Cleric investigation interface for investigation 33721. The interface is divided into several sections:

- Header:** A circular icon with a pin and the text "Investigation: 33721".
- Search/Action Bar:** A search icon, a "Getting context..." status indicator with a starburst icon, and a "Pause" button.
- Issue Details:** A box containing:
 - Issue: Intermittent checkout failures reported by users
 - Duration: 00:27
 - Status: In Progress
- Severity and Owner:** A box containing:
 - Severity: Unknown
 - Owner: Willem Pienaar
- Documents:** A section titled "Documents" with a box containing the text "No documents found yet..".
- Root Cause:** A section titled "Root Cause" which is currently empty.

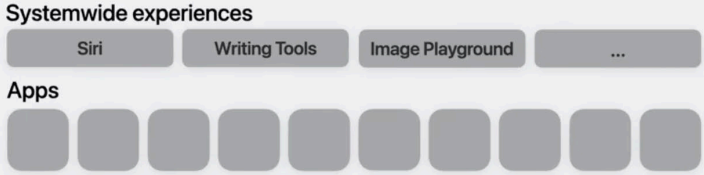
</>

How do we build these types of agents?

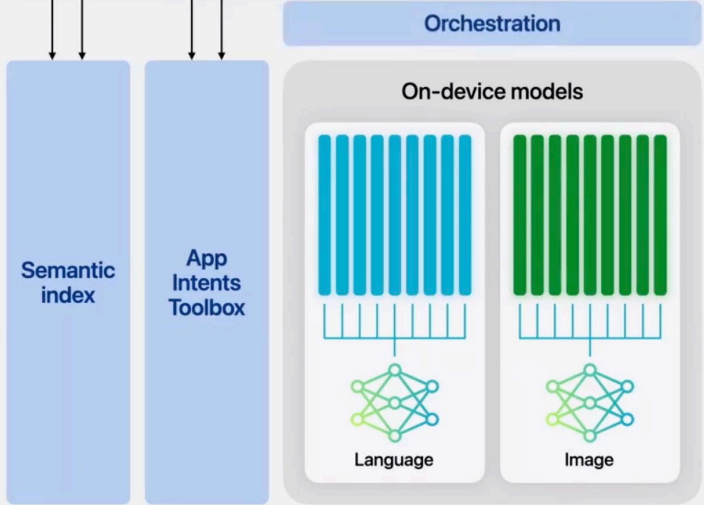
AGENTS ARE SYSTEMS NOT MODELS



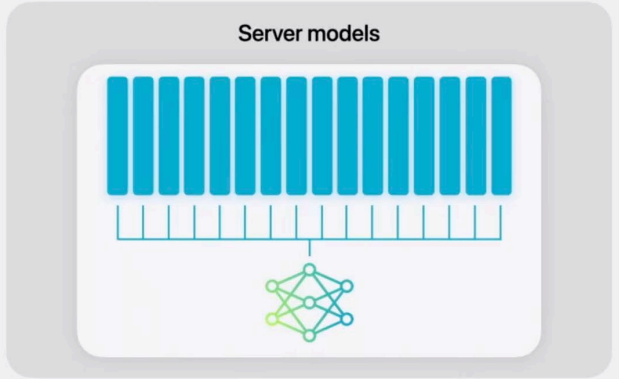
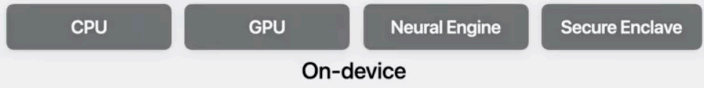
Apps and experiences



Personal Intelligence System



Apple silicon



Private Cloud Compute



</>

Let's talk about models

Open source models are lagging

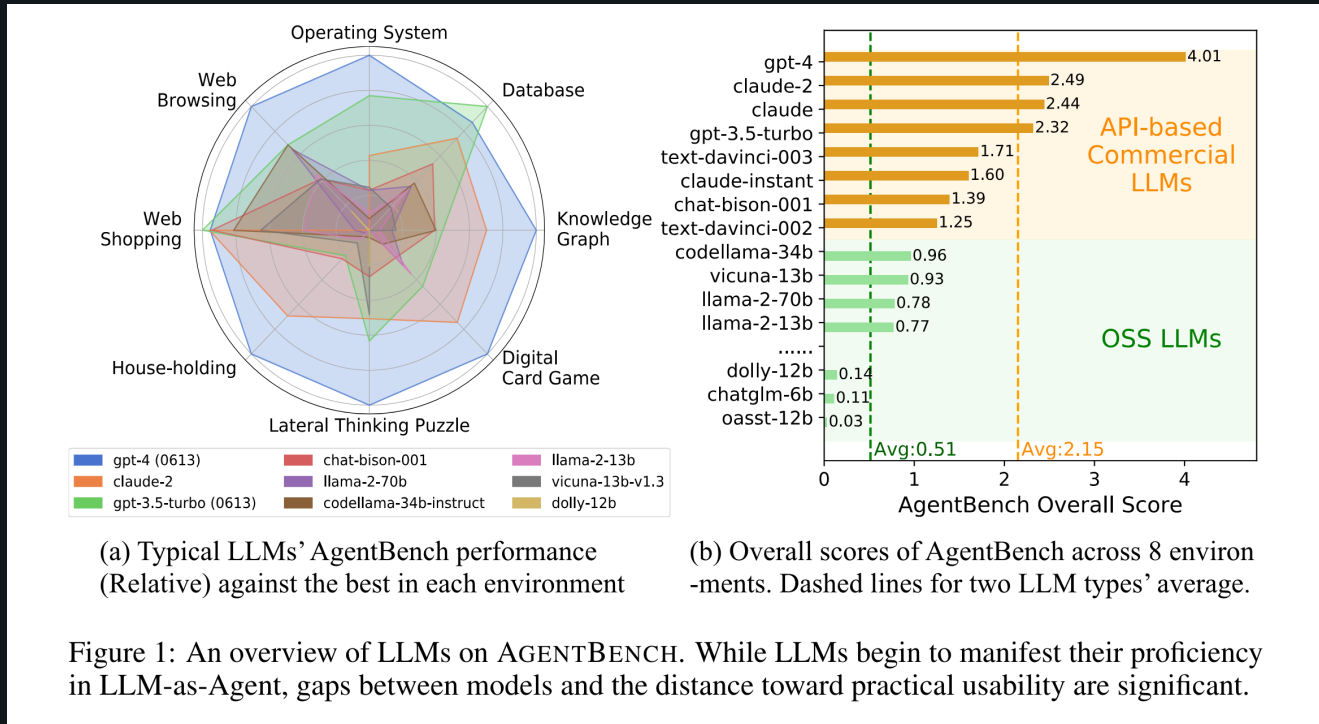


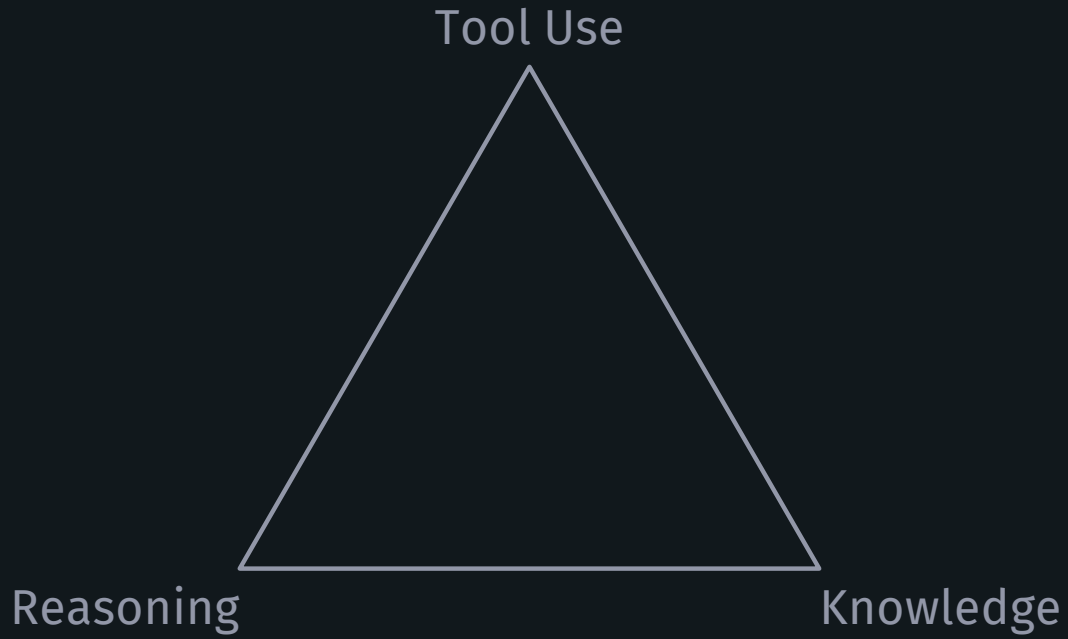
Figure 1: An overview of LLMs on AGENTBENCH. While LLMs begin to manifest their proficiency in LLM-as-Agent, gaps between models and the distance toward practical usability are significant.

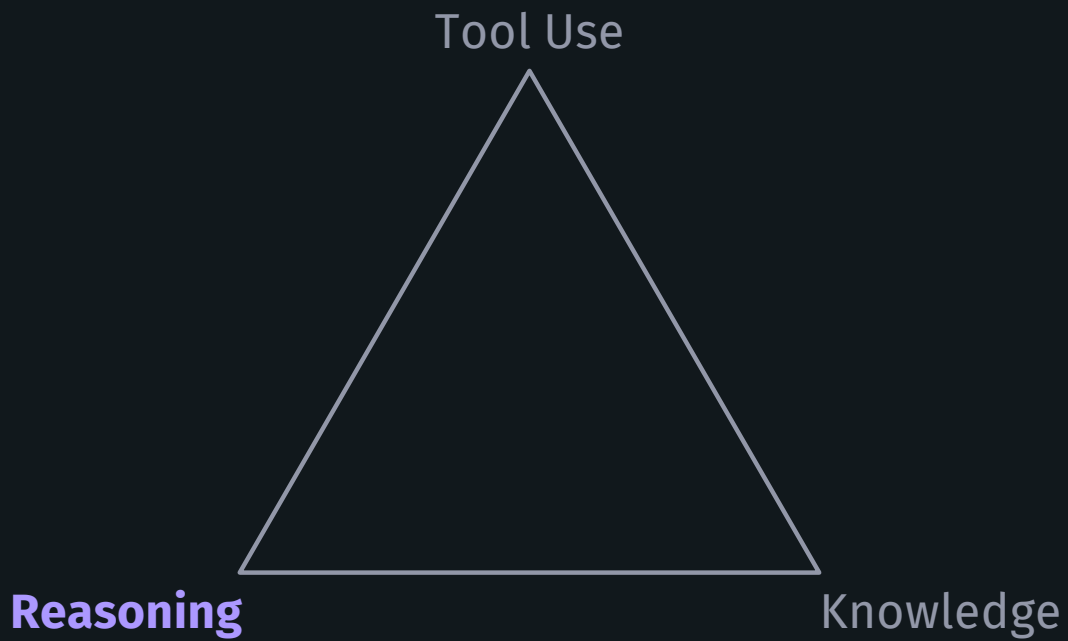
Leaderboard (Lite)

SWE-bench *Lite* is a subset of SWE-bench that's been curated to make evaluation less costly and more accessible. If you'd like to learn more, please read our [blog post](#).

Model	% Resolved	Date	Logs	Trajs	Site	Verified?	Open?
🏆 CodeR + GPT 4 (1106)	28.33	2024-06-04	🔗	-	🔗	✘	✘
🥈 Aider + GPT 4o & Claude 3 Opus	26.33	2024-05-23	🔗	-	🔗	✘	✓
🥉 OpenCSG StarShip CodeGenAgent + GPT 4 (0613)	23.67	2024-05-24	🔗	-	🔗	✘	✘
Moatless Tools + GPT 4o (2024-05-13)	23.33	2024-06-09	🔗	🔗	🔗	✓	✓
Bytedance MarsCode Agent	22.00	2024-05-27	🔗	-	🔗	✘	✘
Amazon Q Developer Agent (v20240430-dev)	20.33	2024-05-09	🔗	-	🔗	✘	✘
AutoCodeRover (v20240408) + GPT 4 (0125)	19.00	2024-05-30	🔗	-	🔗	✘	✓
SWE-agent + GPT 4 (1106)	18.00	2024-04-02	🔗	🔗	🔗	✓	✓
SWE-agent + GPT 4o (2024-05-13)	17.00	2024-06-03	-	-	🔗	✓	✓
SWE-agent + Claude 3 Opus	11.67	2024-04-02	🔗	🔗	-	✓	✓
RAG + Claude 3 Opus	4.33	2024-04-02	🔗	-	🔗	✓	✓
RAG + Claude 2	3.00	2023-10-10	🔗	-	-	✓	✓
RAG + GPT 4 (1106)	2.67	2024-04-02	🔗	-	-	✓	✓
RAG + SWE-Llama 7B	1.33	2023-10-10	🔗	-	-	✓	✓
RAG + SWE-Llama 13B	1.00	2023-10-10	🔗	-	-	✓	✓
RAG + ChatGPT 3.5	0.33	2023-10-10	🔗	-	-	✓	✓

The **% Resolved** metric is out of 300 instances for SWE-bench Lite.





</>

Is there hope?

Table 2: Latency, size, and success rate of TinyAgent models before and after quantization. Latency is the end-to-end latency of the function calling planner, including the prompt processing time and generation.

Model	Weight Precision	Latency (seconds)	Model Size (GB)	Success Rate (%)
GPT-3.5	Unknown	3.2	Unknown	65.04
GPT-4-Turbo	Unknown	3.9	Unknown	79.08
TinyAgent-1.1B	16	3.9	2.2	80.06
TinyAgent-1.1B	4	2.9	0.68	80.35
TinyAgent-7B	16	19.5	14.5	84.95
TinyAgent-7B	4	13.1	4.37	85.14

</>

How do we fine-tune a model for agents?

Choose how you want to call functions

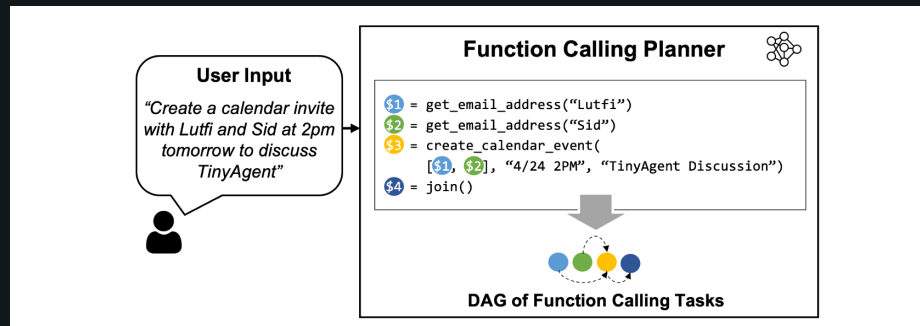
```
from openai import OpenAI
client = OpenAI()

tools = [
  {
    "type": "function",
    "function": {
      "name": "get_current_weather",
      "description": "Get the current weather in a given location",
      "parameters": {
        "type": "object",
        "properties": {
          "location": {
            "type": "string",
            "description": "The city and state, e.g. San Francisco, CA",
          },
          "unit": {"type": "string", "enum": ["celsius", "fahrenheit"]},
        },
        "required": ["location"],
      },
    },
  },
]

messages = [{"role": "user", "content": "What's the weather like in Boston today?"}]
completion = client.chat.completions.create(
  model="gpt-4o",
  messages=messages,
  tools=tools,
  tool_choice="auto"
)
print(completion)

... {
  "index": 0,
  "message": {
    "role": "assistant",
    "content": null,
    "tool_calls": [
      {
        "id": "call_abc123",
        "type": "function",
        "function": {
          "name": "get_current_weather",
          "arguments": "{\\n\\n\"location\\\": \\\"Boston, MA\\\"\\n\\n}"
        }
      }
    ]
  }
}
```

OpenAI



TinyAgent/LLMCompiler

Generate synthetic data using self-instruct or agent gym

```
INVOCATION_FILLER_PROMPT = """
1) Input reasonable values for 'fill_in_string' and 'fill_in_int' in the invocation here: {invocation}
the entire function provided here :(function) to get context over what proper fill_in_string and fill_in_int
Example:

Input: invocation: {{
  "name": "control_camera",
  "arguments": {{
    "mode": "video",
    "duration": "fill_in_int"
  }}
}},
function:(function)

Output: invocation: {{
  "name": "control_camera",
  "arguments": {{
    "mode": "video",
    "duration": 30
  }}
}}

MAKE SURE output is just a dictionary with keys 'name' and 'arguments', no other text or response.

Input: {invocation}
Output:
"""

COMMAND_GENERATION_PROMPT = """
You are to output 2 commands, questions or statements that would generate the inputted function and para
Please make the commands or questions natural, as a person would ask, and the command or questions shou
It should not always mirror the exact technical terminology used in the function and parameters, rather
For instance, the prompt should not be 'turn on the dome light', as that is too technical, but rather 't
Another example, is the prompt should not be 'turn on the HVAC', but rather 'turn on the air conditionin
it is technically incorrect but colloquially used.

RULES: ALWAYS put a backwards slash before an apostrophe or single quote '. For example, do not say don'
Prompts MUST be in double quotes as well.

Example

Input: {{'name': 'calibrate_sensors', 'arguments': {}}}}
Prompt: "The sensors are out of whack, can you reset them", "The calibration of the drone is off, fix i

Input: {{'name': 'set_autopilot', 'arguments': {{'status': 'off'}}}}
Prompt: "OK, I want to take back pilot control now", "Turn off the automatic pilot I'm ready control it"

Input: {invocation}
Prompt:
"""
```

OpenAI Cookbook

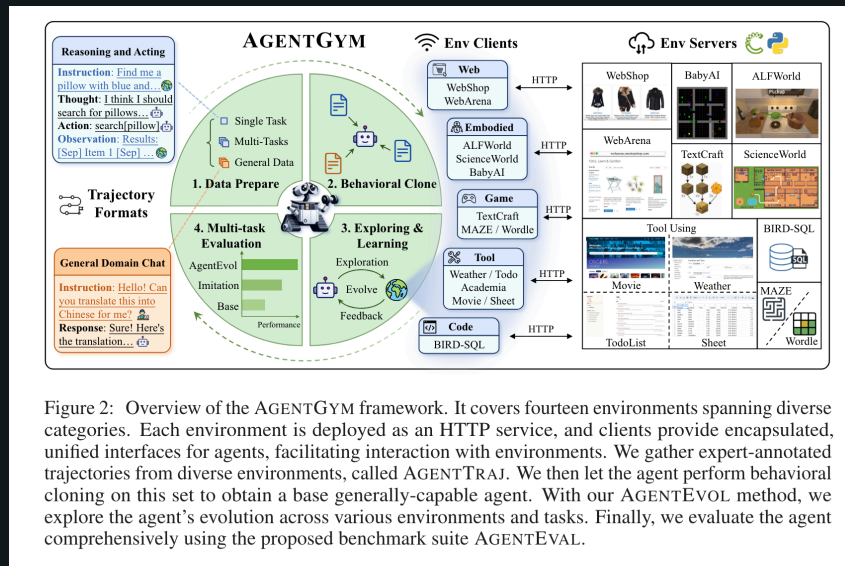
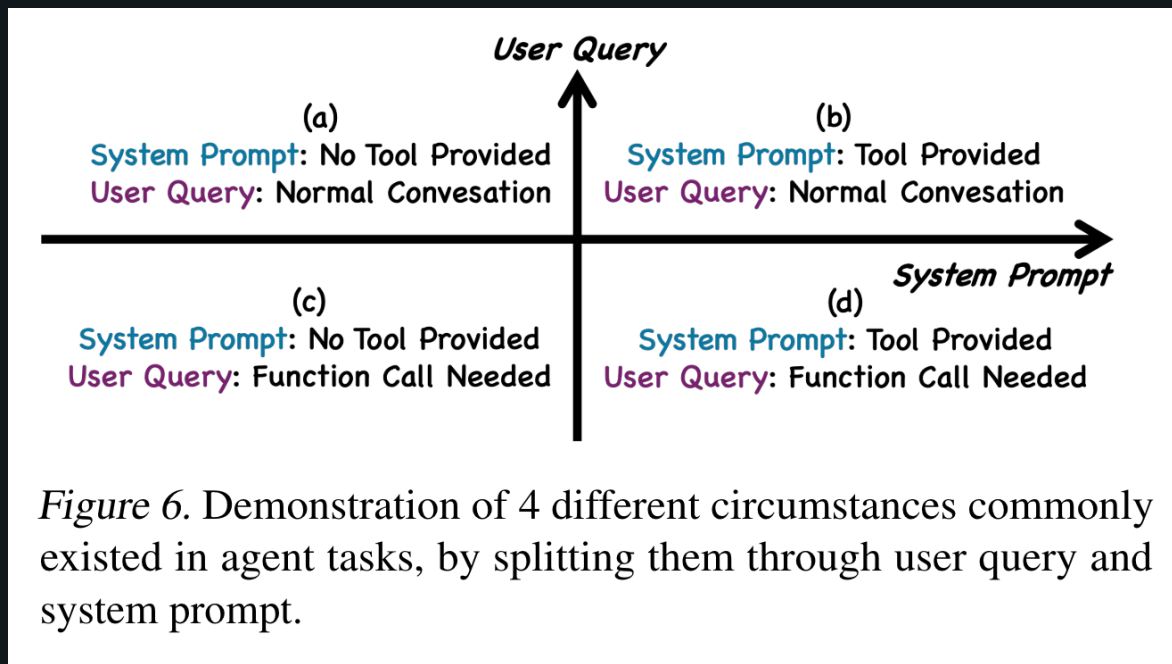


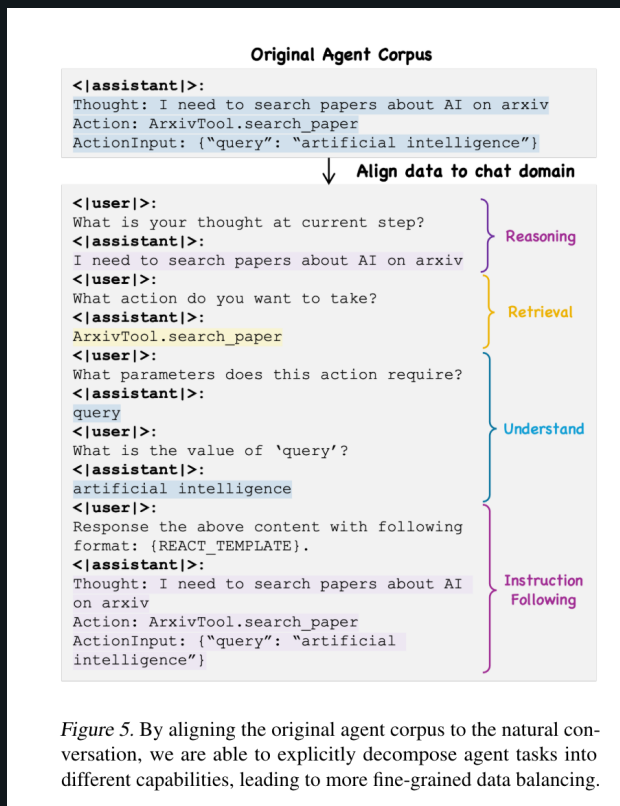
Figure 2: Overview of the AGENTGYM framework. It covers fourteen environments spanning diverse categories. Each environment is deployed as an HTTP service, and clients provide encapsulated, unified interfaces for agents, facilitating interaction with environments. We gather expert-annotated trajectories from diverse environments, called AGENTTRAJ. We then let the agent perform behavioral cloning on this set to obtain a base generally-capable agent. With our AGENTEVOL method, we explore the agent's evolution across various environments and tasks. Finally, we evaluate the agent comprehensively using the proposed benchmark suite AGENTEVOL.

AgentGym

Don't forget to cover the full scope of user behavior



Choose chat template suitable for LLM training



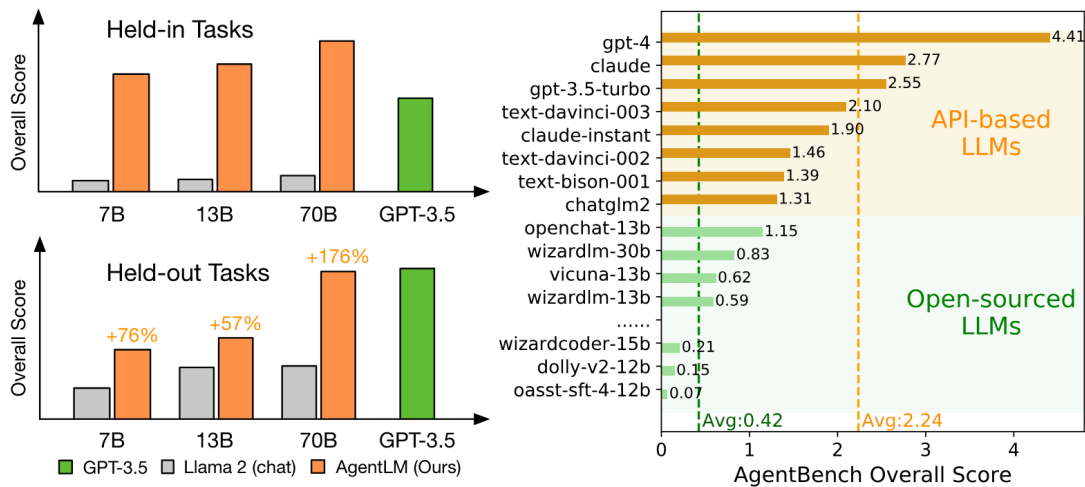
← ReAct chat template

← Agent-FLAN chat template

Figure 5. By aligning the original agent corpus to the natural conversation, we are able to explicitly decompose agent tasks into different capabilities, leading to more fine-grained data balancing.

Agent-FLAN: <https://arxiv.org/abs/2403.12881>

Fine-tune your base model



(a) Overall score in our held-in and held-out tasks. (b) Closed & open LLMs on agent tasks (Liu et al., 2023)

Figure 1: (a) **AgentLM exhibits superior performance.** AgentLM is a series of models fine-tuned on the foundation of Llama 2 chat. Moreover, its generalization capability on held-out tasks is on par with GPT-3.5; (b) This figure is directly re-printed from AgentBench (Liu et al., 2023) with permission. **Open LLMs significantly underperforms API-based LLMs.**

Consider embedding within a multi-agent system

Table 2: Results on AlpacaEval 2.0 and MT-Bench. For AlpacaEval 2.0, MoA and MoA-Lite correspond to the 6 proposer with 3 layers and with 2 layer respectively. MoA w/ GPT-4o corresponds to using GPT-4o as the final aggregator in MoA. We ran our experiments three times and reported the average scores along with the standard deviation. [†] denotes our replication of the AlpacaEval results. We ran all the MT-Bench scores ourselves to get turn-based scores.

(a) AlpacaEval 2.0			(b) MT-Bench.			
Model	LC win.	win.	Model	Avg.	1st turn	2nd turn
MoA w/ GPT-4o	65.7 \pm 0.7%	78.7 \pm 0.2%	MoA w/ GPT-4o	9.40 \pm 0.06	9.49	9.31
MoA	65.1 \pm 0.6%	59.8 \pm 0.3%	GPT-4 Turbo (04/09)	9.31	9.35	9.28
MoA-Lite	59.3 \pm 0.2%	57.0 \pm 0.7%	MoA	9.25 \pm 0.10	9.44	9.07
GPT-4 Omni (05/13)	57.5%	51.3%	GPT-4 Preview (11/06)	9.20	9.38	9.03
GPT-4 Turbo (04/09)	55.0%	46.1%	GPT-4 Omni (05/13)	9.19	9.31	9.07
WizardLM 8x22B [†]	51.3%	62.3%	MoA-Lite	9.18 \pm 0.09	9.38	8.99
GPT-4 Preview (11/06)	50.0%	50.0%	Qwen1.5 110B Chat	8.96	9.23	8.63
Qwen1.5 110B Chat	43.9%	33.8%	Llama 3 70B Instruct	8.94	9.2	8.68
Qwen1.5 72B Chat	36.6%	26.5%	Mixtral 8x22B v0.1	8.78	9.11	8.44
GPT-4 (03/14)	35.3%	22.1%	WizardLM 8x22B	8.78	8.96	8.61
Llama 3 70B Instruct	34.4%	33.2%	Qwen1.5 72B Chat	8.44	8.55	8.34
Mixtral 8x22B v0.1	30.9%	22.2%	GPT-4 (06/13)	8.84	9.08	8.61

Should you fine-tune open source models for AI agents?

CON

- Proprietary models are still significantly ahead of open source models for agent use cases.
- Collecting agent trajectories for complex tasks like coding is non-trivial.
- Fine-tuning can easily degrade general performance and become a game of whack-a-mole.
- Scale effects are very real. You're unlikely to beat frontier models for generalist agents.

PRO

- You will learn a lot and generate a lot of useful data.
- Self-instruct and agent gyms makes collection of trajectories feasible for many use cases.
- You can significantly increase performance of agents in specific domains, even beating frontier models.
- Open source frontier models are getting better just like proprietary models.
- It allows you to control your own destiny.



Learn how and give it a shot.

</>

Thank you

<https://continual.ai>
tristan@continual.ai