

# Executable Code Actions Elicit Better LLM Agents

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, Heng Ji  
{xingyao6, yangyi3, lifan4, yunzhuli, haopeng, hengji}@illinois.edu, yizzhang@apple.com



## Motivation

- Large Language Model (LLM) agents, capable of performing a broad range of actions, such as invoking tools and controlling robots, show great potential in tackling real-world challenges.
- LLM agents are typically prompted to produce actions by generating **JSON** or **text** in a pre-defined format, usually limited by constrained action space (e.g., the scope of pre-defined tools) and restricted flexibility (e.g., inability to compose multiple tools)

This work proposes to use **executable code** to consolidate LLM agents' actions into a unified action space (**CodeAct**).

## CodeAct Framework

CodeAct is a general-purpose framework that allows LLMs to generate **executable Python code as actions**.

Table 1: The benefit of CodeAct compared to using Text/JSON for LLM action.

|   | CodeAct for LLM action   | JSON or Text for LLM action   |
|---|--|---|
| <b>Availability of Data</b>   | ✓ Large quantity of code available <sup>1</sup> for pre-training   | ✗ Data curation required for particular format  |
| <b>Complex Operation</b> (e.g., looping, composition of multiple tools) | ✓ Natively supported via control and data flow   | ✗ Requires careful engineering if feasible (e.g., define new tools to mimic if-statement)   |
| <b>Availability of Tools</b>  | ✓ Can directly use existing software packages <sup>2</sup>   | ✗ Requires human effort to curate tools from scratch or existing software   |
| <b>Automated Feedback</b>   | ✓ Feedback mechanism <sup>3</sup> (e.g., traceback) is already implemented as an infrastructure for most programming languages | ✗ Requires human effort to provide feedback or re-route feedback from the underlying programming language used to implement the tools |

<sup>1</sup> Including code demonstrating useful behaviors for LLM agents (e.g., task decomposition, coordination of multiple function calls to different tools).  
<sup>2</sup> Human-written Python packages covering a wide range of applications are available on <https://pypi.org/>.  
<sup>3</sup> For example, in Python, errors and exceptions (<https://docs.python.org/3/tutorial/errors.html>) are available. Most software provides error messages in natural language to help human programmers debug their code. CodeAct enables LLM to use them directly.

Comparing **CodeAct** with Text/JSON: It gets more thing done in fewer turns of interactions - up to 20% higher success rate w/ 30% fewer turns on M3ToolEval

**Instruction:** Determine the most cost-effective country to purchase the smartphone model "CodeAct 1". The countries to consider are the USA, Japan, Germany, and India.

**Available APIs**  
(1) lookup\_rates(country: str) -> (float, float)  
(2) convert\_and\_tax(price: float, exchange\_rate: float, tax\_rate: float) -> float  
(3) estimate\_final\_price(converted\_price: float, shipping\_cost: float) -> float  
(4) lookup\_phone\_price(model: str, country: str) -> float  
(5) estimate\_shipping\_cost(destination\_country: str) -> float

**LLM Agent using [Text/JSON] as Action**

Think: I should calculate the phone price in USD for each country, then find the most cost-effective country.

Action: Text: lookup\_rates, Germany  
JSON: {"tool": "lookup\_rates", "country": "Germany"}

Environment 1.1, 0.19

Action: Text: lookup\_phone\_price, CodeAct 1, Germany  
JSON: {"tool": "lookup\_phone\_price", "model": "CodeAct 1", "country": "Germany"}

Environment 788

Action: Text: convert\_and\_tax, 788, 1.1, 0.19  
JSON: {"tool": "convert\_and\_tax", "price": 788, "exchange\_rate": 1.1, "tax\_rate": 0.19}

Environment 916.3

[... interactions omitted (look up shipping cost and calculate final price) ...]

Action: Text: lookup\_rates, Japan  
JSON: {"tool": "lookup\_rates", "country": "Japan"}

[... interactions omitted (calculate final price for all other countries) ...]

Response: The most cost-effective country to purchase the smartphone model is Japan with price 984.88 in USD.

**CodeAct: LLM Agent using [Code] as Action**

Think: I should calculate the phone price in USD for each country, then find the most cost-effective country.

Action: Control & Data Flow of Code Simplifies Complex Operations

```
countries = ['USA', 'Japan', 'Germany', 'India']
final_prices = {}

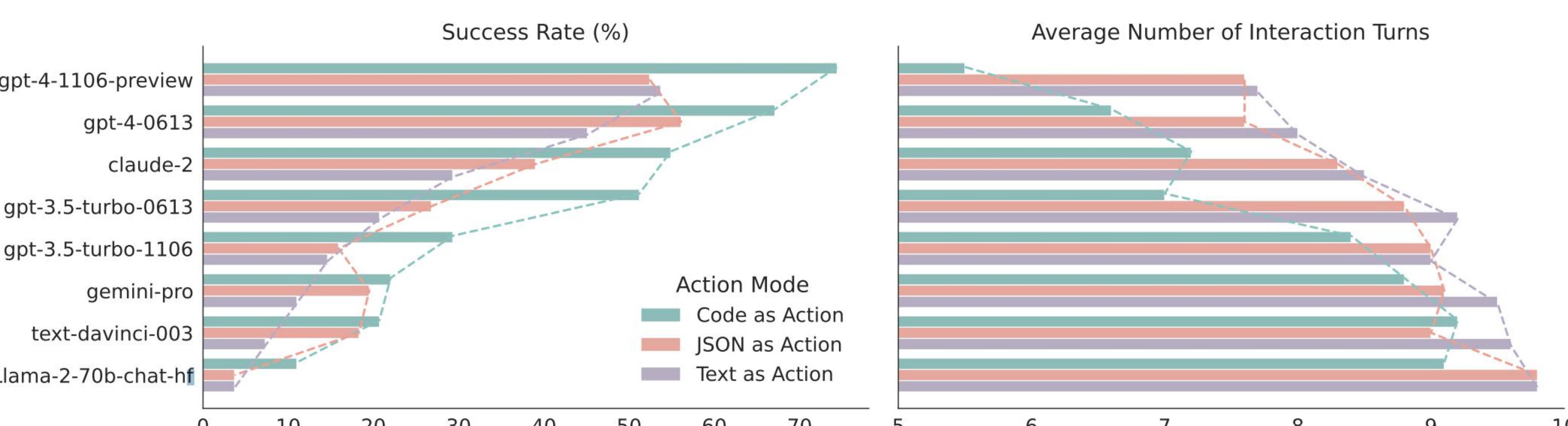
for country in countries:
    exchange_rate, tax_rate = lookup_rates(country)
    local_price = lookup_phone_price("CodeAct 1", country)
    converted_price = convert_and_tax(
        local_price, exchange_rate, tax_rate)
    shipping_cost = estimate_shipping_cost(country)
    final_price = estimate_final_price(converted_price, shipping_cost)
    final_prices[country] = final_price

most_cost_effective_country = min(final_prices, key=final_prices.get)
most_cost_effective_price = final_prices[most_cost_effective_country]
print(most_cost_effective_country, most_cost_effective_price)
```

Re-use 'min' Function from Existing Software Infrastructures (Python library)

Environment 1.1, 0.19

Response: The most cost-effective country to purchase the smartphone model is Japan with price 984.88 in USD.



## Advantage of CodeAct

**CodeAct Shows the Promise as a Strong Tool Use Framework (Table 2)** Compared to Text/JSON, CodeAct achieves comparable or better performance even in *atomic* actions (the simplistic tool use scenario) where its control and data flow strengths are ablated.

Table 2: Atomic API call correctness on API-Bank. The best performance is **bolded**, and the second-best is underlined.

| Format of Action          | Correctness (% ↑) |      |      |
|---------------------------|-------------------|------|------|
|                           | CodeAct           | JSON | Text |
| <b>Open-source LLMs</b>   |                   |      |      |
| CodeLlama-7b-Instruct-hf  | 12.5              | 12.0 | 17.0 |
| CodeLlama-13b-Instruct-hf | 11.8              | 7.8  | 14.0 |
| CodeLlama-34b-Instruct-hf | 17.3              | 12.0 | 16.8 |
| Llama-2-7b-chat-hf        | 28.8              | 11.3 | 25.8 |
| Llama-2-13b-chat-hf       | 38.1              | 8.5  | 37.3 |
| Llama-2-70b-chat-hf       | 35.6              | 14.3 | 37.6 |
| Mistral-7B-Instruct-v0.1  | 2.5               | 2.3  | 3.0  |
| lemur-70b-chat-v1         | 58.6              | 46.6 | 56.1 |
| <b>Closed-source LLMs</b> |                   |      |      |
| claude-2                  | 76.7              | 59.4 | 73.2 |
| claude-instant-1          | 75.2              | 64.9 | 73.7 |
| geminipro                 | 70.4              | 73.2 | 71.2 |
| gpt-3.5-turbo-0613        | 74.4              | 73.9 | 73.4 |
| gpt-3.5-turbo-1106        | 75.4              | 78.4 | 74.4 |
| gpt-4-0613                | 75.4              | 82.0 | 74.4 |
| gpt-4-1106-preview        | 76.7              | 82.7 | 73.4 |
| text-davinci-002          | 69.2              | 59.6 | 57.4 |
| text-davinci-003          | 75.4              | 76.9 | 69.7 |

Table 3: Success rates (higher the better) and average turns required per instance (lower the better) on M<sup>3</sup>ToolEval. The best results for each model are **bolded**, and the second-best ones are underlined.

| Format of Action          | Success Rate (% ↑) |      |      | Avg. Turns (↓) |      |      |
|---------------------------|--------------------|------|------|----------------|------|------|
|                           | CodeAct            | JSON | Text | CodeAct        | JSON | Text |
| <b>Open-source LLMs</b>   |                    |      |      |                |      |      |
| CodeLlama-7b-Instruct-hf  | 4.9                | 2.4  | 2.4  | 9.7            | 9.9  | 9.9  |
| CodeLlama-13b-Instruct-hf | 4.9                | 4.9  | 4.9  | 9.8            | 9.8  | 9.7  |
| CodeLlama-34b-Instruct-hf | 2.4                | 0.0  | 0.0  | 9.9            | 10.0 | 10.0 |
| Llama-2-7b-chat-hf        | 0.0                | 1.2  | 2.4  | 8.9            | 9.5  | 9.6  |
| Llama-2-13b-chat-hf       | 0.0                | 0.0  | 0.0  | 9.7            | 10.0 | 10.0 |
| Llama-2-70b-chat-hf       | 11.0               | 3.7  | 3.7  | 9.1            | 9.8  | 9.8  |
| Mistral-7B-Instruct-v0.1  | 0.0                | 3.7  | 1.2  | 10.0           | 9.8  | 9.9  |
| lemur-70b-chat-v1         | 13.4               | 15.9 | 12.2 | 9.1            | 9.3  | 9.4  |
| <b>Closed-source LLMs</b> |                    |      |      |                |      |      |
| claude-2                  | 54.9               | 39.0 | 29.3 | 7.2            | 8.3  | 8.5  |
| claude-instant-1          | 20.7               | 31.7 | 24.4 | 8.8            | 8.6  | 8.9  |
| geminipro                 | 22.0               | 19.5 | 11.0 | 8.8            | 9.1  | 9.5  |
| gpt-3.5-turbo-0613        | 51.2               | 26.8 | 20.7 | 7.0            | 8.8  | 9.2  |
| gpt-3.5-turbo-1106        | 29.3               | 15.9 | 14.6 | 8.4            | 9.0  | 9.0  |
| gpt-4-0613                | 67.1               | 56.1 | 45.1 | 6.6            | 7.6  | 8.0  |
| gpt-4-1106-preview        | 74.4               | 52.4 | 53.7 | 5.5            | 7.6  | 7.7  |
| text-davinci-002          | 4.9                | 4.9  | 8.5  | 9.7            | 9.8  | 9.6  |
| text-davinci-003          | 20.7               | 18.3 | 7.3  | 9.2            | 9.0  | 9.6  |

## CodeAct Gets More Done with Fewer Interactions (Table 3)

CodeAct generally has a higher task success rate (12 out of 17 evaluated LLMs). Moreover, using CodeAct requires a lower average number of turns (12 out of 17 evaluated LLMs).

The ability to benefit from **multi-turn interactions** and **existing software packages**. Thanks to LLM's extensive knowledge of Python learned during pre-training, the LLM agent can automatically import the correct Python libraries to solve tasks without requiring user-provided tools or demonstrations.

**User**: Here is a dataset about auto MPG: "https://buggingface.com/datasets/scikit-learn/autos-mpg/raw/main/autos-mpg.csv". You should download and load the dataset from the URL, take a look at it, check for missing values, do a train-test split, and train a regression model predicting its MPG. You should use everything except "mpg" and "car name" as the input features.

**CodeActAgent**: To accomplish this task, I will use the pandas library to download and load the dataset, check for missing values, and perform a train-test split. Then, I will use the scikit-learn library to train a regression model.

**User**: Can you also give me the MSE and R<sup>2</sup> for the training set?

**CodeActAgent**: Certainly! Here is the mean squared error (MSE) and R<sup>2</sup> score for the training set: Mean squared error: 0.10711 and the R<sup>2</sup> score is 0.790, which indicates that the model has a good fit with the data [...]

**User**: Can you also give me the MSE and R<sup>2</sup> for the training set?

**CodeActAgent**: Certainly! Here is the mean squared error (MSE) and R<sup>2</sup> score for the training set: Mean squared error: 0.10711 and the R<sup>2</sup> score is 0.790, which indicates [...]

**User**: Can you visualize the regression coefficient? You may access them via model.coef\_ and their names via model.feature\_names\_in\_ [...]

**CodeActAgent**: Certainly! Here is a visualization of the regression coefficients [...]

**User**: Can you visualize the regression coefficient? You may access them via model.coef\_ and their names via model.feature\_names\_in\_ [...]

**CodeActAgent**: Certainly! Here is a visualization of the regression coefficients [...]

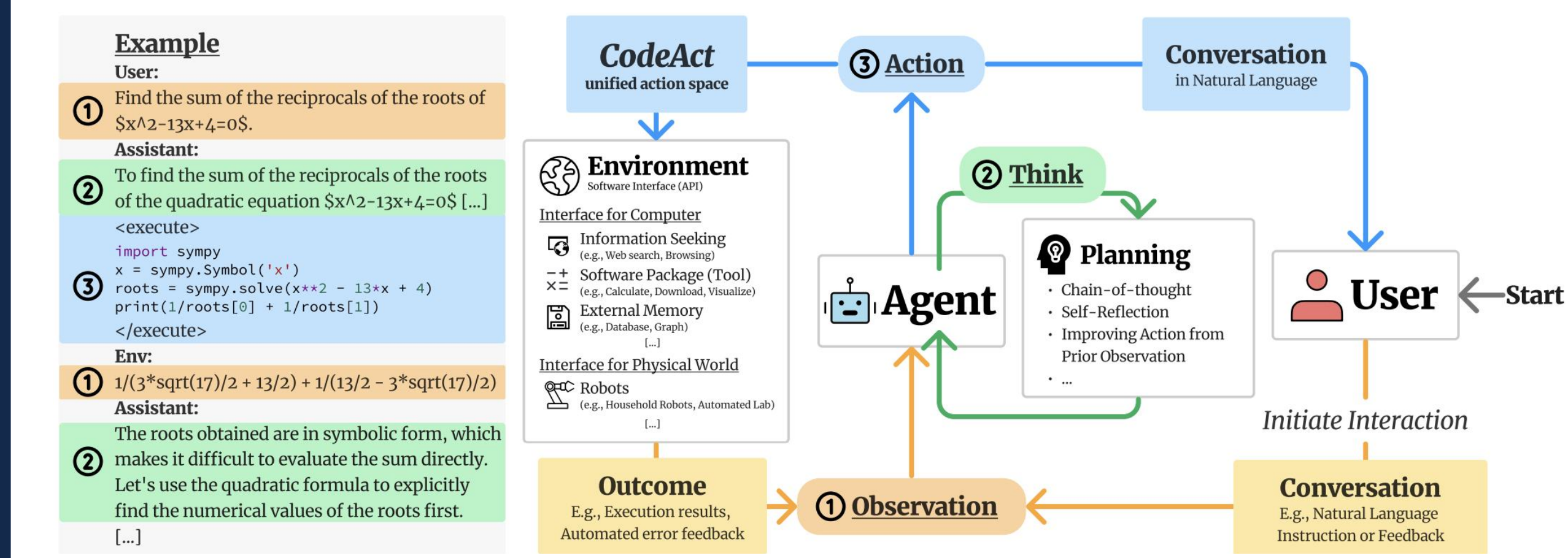
**User**: Can you visualize the regression coefficient? You may access them via model.coef\_ and their names via model.feature\_names\_in\_ [...]

**CodeActAgent**: Certainly! Here is a visualization of the regression coefficients [...]

Using CodeActAgent (based on fine-tuned Mistral-7b). No in-context demonstrations are provided to the model. Some messages are omitted for space. See <https://chat.xwang.dev/r/Vqn108G> for complete interaction.

## Empowering Open-source LLM Agent to be Better at CodeAct

Based on CodeAct framework, we build a **CodeActInstruct** dataset to improve open-source LLM agents, covering Agent-Environment Interactions spanning multiple domains.



- We generate data by running stronger LLMs (e.g., gpt-3.5, gpt-4, claude) to interact with the computer (information seeking, software package tool use, external memory) and the physical world (robot planning).
- To retain the ability to engage in NL conversation, we mix multi-turn human-LLM natural language conversations with CodeActInstruct for LLM supervised fine-tuning (SFT).

| Data Mixture           | Data Type                | Data Name                                   | # of Data Instances | # of Total Tokens | Avg. Tokens Per Instance |
|------------------------|--------------------------|---|---------------------|-------------------|--------------------------|
| Prior Work             | -                        | FireAct (Chen et al., 2023a)                | 2,063               | 542,176           | 262.81                   |
|                        | -                        | AgentInstruct (Zeng et al., 2023)           | 1,866               | 2,517,785         | 1349.30                  |
|                        | Information Seeking      | HotpotQA (Yang et al., 2018)                | 1,664               | 2,472,227         | 1485.71                  |
|                        | Software Packages (Tool) | MATH (Math, (Hendrycks et al., 2021b))      | 1,732               | 1,719,467         | 992.76                   |
|                        | Software Packages (Tool) | APPS (Code, (Hendrycks et al., 2021a))      | 647                 | 1,235,472         | 1909.54                  |
| CodeActInstruct (Ours) | External Memory          | WikiTableQuestion (Pasupat & Liang, 2015)   | 1,065               | 1,316,246         | 1235.91                  |
|                        | Robot Planning           | ALFWorld (Shridhar et al., 2020)            | 2,031               | 3,838,269         | 1889.84                  |
|                        | <b>Total</b>             | <b>Total</b>                                | <b>7,139</b>        | <b>10,581,681</b> | <b>1482.24</b>           |
| General Conversation   | Single-Turn Reasoning    | OpenOrca (Sub-sampled, (Lian et al., 2023)) | 50,000              | 14,034,152        | 280.68                   |
|                        | Multi-Turn Conversations | ShareGPT (Sub-sampled, (Anonymous, 2023))   | 10,000              | 17,933,861        | 1793.39                  |
|                        | Multi-Turn Conversations | ShareGPT (GPT-4, (OpenChat, 2023))          | 4,583               | 18,195,878        | 3970.30                  |
|                        | Multi-Turn Reasoning     | Capybara (LDJr, 2023)                       | 4,647               | 4,982,435         | 1072.18                  |
|                        | <b>Total</b>             | <b>Total</b>                                | <b>69,230</b>       | <b>55,146,326</b> | <b>796.57</b>            |

We train CodeActAgent on CodeActInstruct and general conversations with two variants of backbone LLM (Llama-2 7B and Mistral-7b).

- We find the resulting model **excel in agent task that uses code as action** (both in-domain and out-of-domain).
- Such superior performance also **generalize to out-of-domain text as action**.
- These agent capabilities does not come at the cost of the model's general knowledge, coding, reasoning, and natural language dialog capability!

| Model                                   | Size | Agent Tasks    |                     |                              | Generic Tasks (OD) |      |           | Overall Average |       |         |      |
|---|------|----------------|---------------------|------------------------------|--------------------|------|-----------|-----------------|-------|---------|------|
|   |      | Code as Action | Text as Action (OD) | M <sup>3</sup> ToolEval (OD) | Miniwob++ SciWorld | MMLU | HumanEval |                 | GSM8K | MTBench |      |
| <b>Open-source LLMs (LLaMA-2-based)</b> |      |                |                     |                              |                    |      |           |                 |       |         |      |
| Llama2 Chat                             | 7B   | -              | -                   | -                            | -                  | 45.3 | 12.8      | 14.6            | -     | -       |      |
| Llama2 Base                             | 7B   | 3.2            | 11.0                | 0.0                          | 0.0                | 5.9  | 48.0      | 13.9            | 27.7  | 6.3     | 21.1 |
| FireAct (Chen et al., 2023a)            | 7B   | 0.0            | 0.3                 | 0.0                          | 0.0                | 6.8  | 44.1      | 3.5             | 12.4  | 4.5     | 14.0 |
| CodeActAgent (LLaMA-2)                  | 7B   | 8.7            | 6.1                 | 0.0                          | 28.9               | 13.7 | 48.7      | 15.4            | 24.6  | 6.1     | 24.8 |
| CodeActAgent (LLaMA-2)                  | 7B   | 51.3           | 20.4                | 0.0                          | 25.5               | 17.6 | 50.6      | 18.1            | 38.3  | 7.5     | 30.7 |
| <b>Open-source LLMs (Mistral-based)</b> |      |                |                     |                              |                    |      |           |                 |       |         |      |
| Mistral Base                            | 7B   | -              | -                   | -                            | -                  | 60.1 | 30.5      | 52.1            | -     | -       | 25.6 |
| Mistral Instruct                        | 7B   | 18.8           | 9.7                 | 0.0                          | 0.5                | 4.0  | 53.8      | 29.3            | 43.3  | 6.4     | 25.6 |
| CodeActAgent (Mistral)                  | 7B   | 57.4           | 32.4                | 12.2                         | 46.2               | 15.9 | 59.1      | 34.7            | 58.0  | 8.2     | 42.5 |
| <b>Closed-source LLMs</b>               |      |                |                     |                              |                    |      |           |                 |       |         |      |
| gpt-3.5-turbo-0613                      | -    | 33.9           | 38.2                | 51.2                         | 66.7               | 21.2 | 70.0      | 48.1            | 57.1  | 7.9     | 54.0 |
| gpt-4-0613                              | -    | 68.6           | 70.2                | 67.1                         | 69.4               | 36.4 | 86.4      | 67.0            | 87.1  | 9.0     | 71.7 |

Some results are only available with instruction-tuned models.

**Paper** **Code** **Try out CodeAct in OpenDevin →**