Spider2-V: How Far Are Multimodal Agents From Automating Data Science and Engineering Workflows?

Ruisheng Cao, Fangyu Lei, Haoyuan Wu, Jixuan Chen, Yeqiao Fu, Hongcheng Gao, Xinzhuang Xiong, Hanchong Zhang, Yuchen Mao, Wenjing Hu, Tianbao Xie, Hongsheng Xu, Danyang Zhang, Sida Wang, Ruoxi Sun, Pengcheng Yin, Caiming Xiong, Ansong Ni, Qian Liu, Victor Zhong, Lu Chen, Kai Yu, Tao Yu

XLANG Lab

香港大學自然語言處理實驗室
Natural Language Processing Group, The University of Hong Kong

SJTU Cross Media Language Intelligence Lab
上海交通大學跨媒體語言智能實驗室

Data Science and Engineering

1. Extended Data Pipeline

- Academia
  - data query -> Spider1.0

What is the average life expectancy in the countries where English is not the official language?

```
SELECT AVG(life_expectancy)
FROM country
WHERE name NOT IN
   (SELECT T1.name
    FROM country AS T1 JOIN
    country_language AS T2
    ON T1.code = T2.country_code
    WHERE T2.language = "English"
      AND T2.is_official = "T")
```

  - data analysis -> DS-1000

Problem

Here is a sample dataframe:

```
df = pd.DataFrame({"A": [1, 2, 3], "B": [4, 5, 6]})
```

I'd like to add inverses of each existing column to the dataframe and name them based on existing column names with a prefix, e.g. inv_A is an inverse of column A and so on.

The resulting dataframe should look like so:

```
result = pd.DataFrame({"A": [1, 2, 3], "B": [4, 5, 6], "inv_A": [1/1, 1/2, 1/3], "inv_B": [1/4, 1/5, 1/6]})
```

Obviously there are redundant methods like doing this in a loop, ***but** there should exist much more pythonic ways of doing it* … [omitted for brevity]

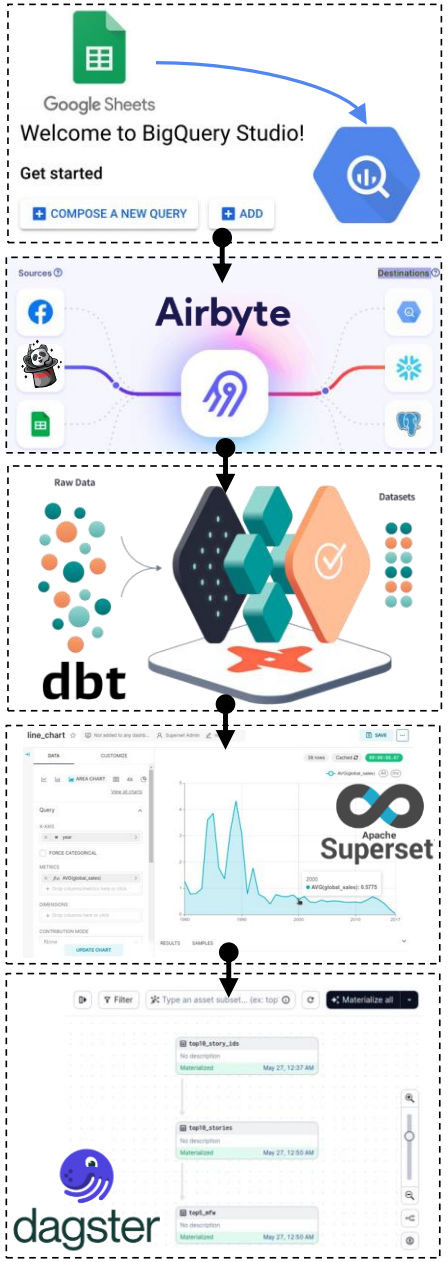
Code Context

```
A:
<code>
import pandas as pd
df = pd.DataFrame({"A": [1, 2, 3],"B": [4, 5, 6]})
</code>
BEGIN SOLUTION
<code>
[insert]
</code>
END SOLUTION
<code>
print(result)
</code>
```

Reference Solution

```
result = df.join(df.apply(lambda x: 1/x).add_prefix("inv_"))
```

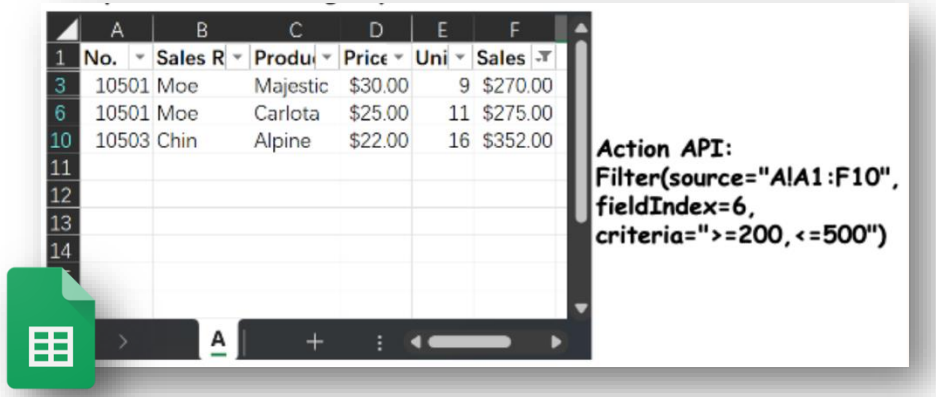
- Industry
  - data warehousing
  - data ingestion
  - data transformation
  - data visualization
  - data orchestration



Data Science and Engineering

2. Professional Enterprise Software

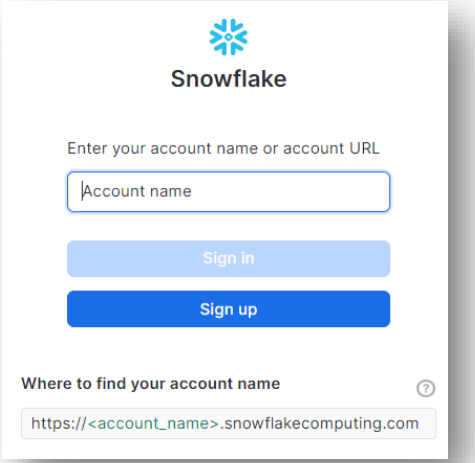
- Academia
  - daily life application -> SheetCopilot

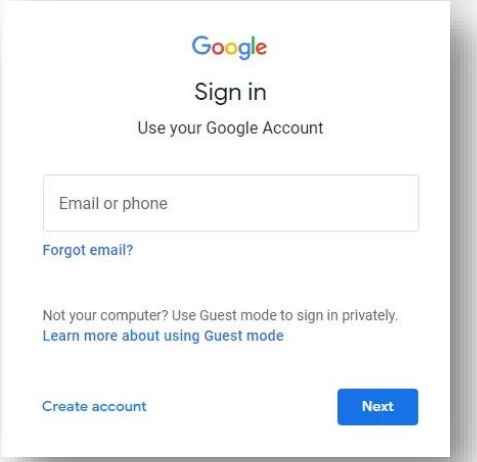


  - common Python libraries -> ARCADE

NumPy scikit learn pandas

- Industry
  - User accounts





snowflake Google BigQuery

data warehousing

Airbyte

data ingestion

dbt Cloud™

data transformation

Apache Superset Metabase

data visualization

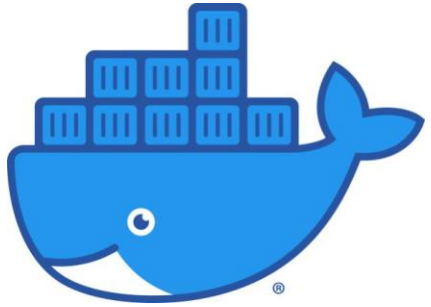
Apache Airflow dagster

data orchestration

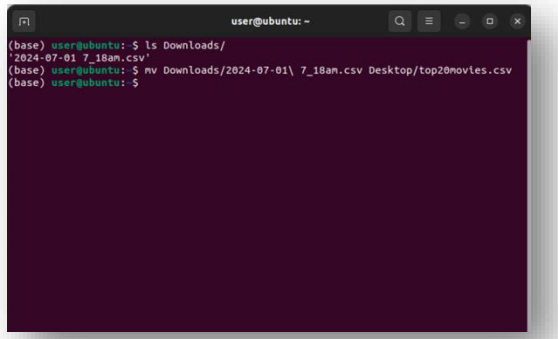
Data Science and Engineering

3. Integrated CLI+GUI Actions

- Academia
  - CLI interface -> Intercode



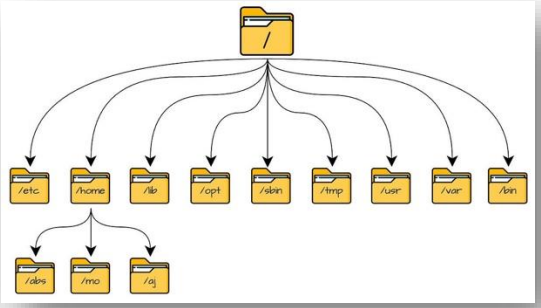
docker



bash

```python
import pandas as pd


def groundbreaking_stuff(data: pd.DataFrame, breaking: bool) -> pd.DataFrame:
    """Groundbreaking stuff done to the data
    Args:
        data (pd.DataFrame): Input Data
    Returns:
        pd.DataFrame: Output Data
    """

    grounded_data = data.ground()

    if breaking:
        return grounded_data.breaking()

    return grounded_data
```

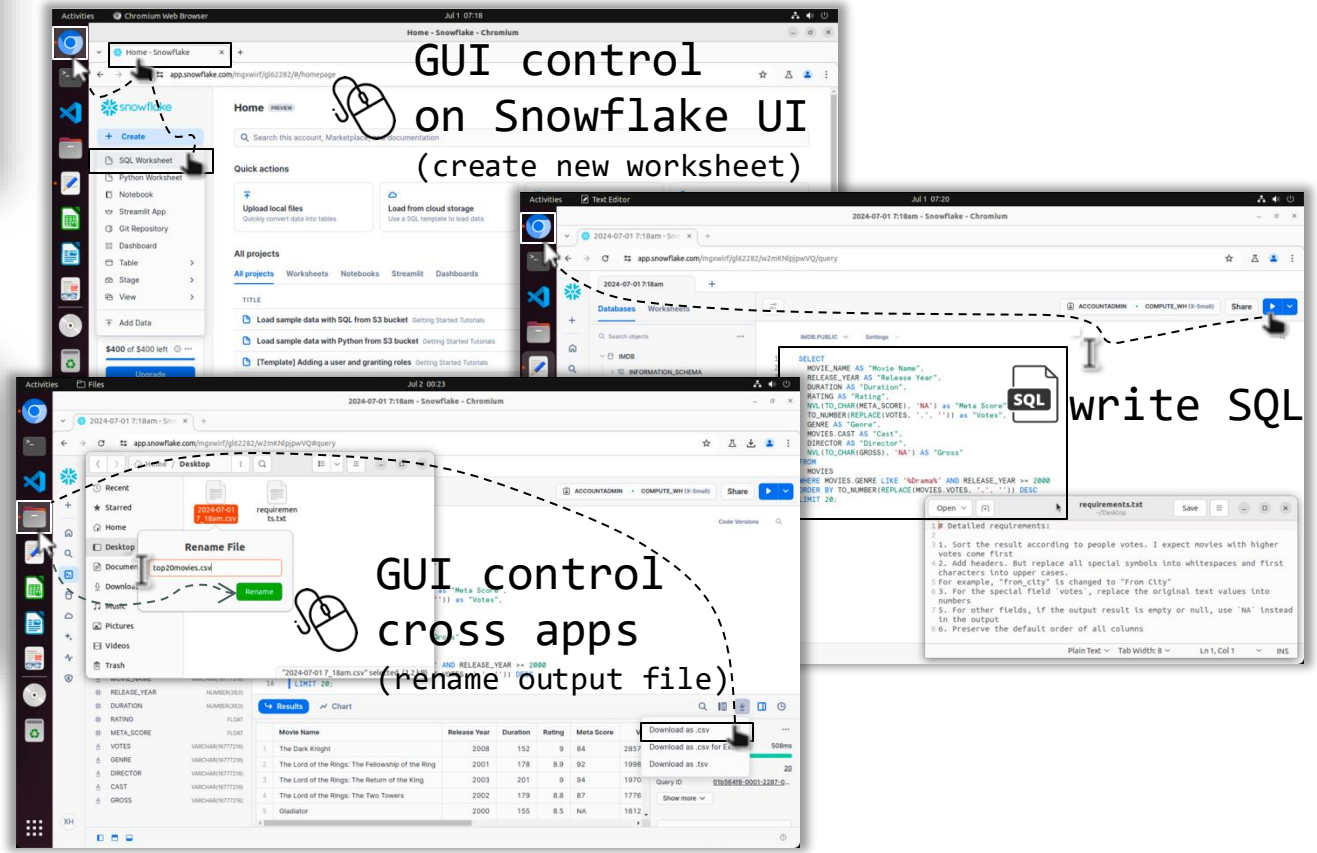
Python interpreter



folder structure

- Industry
  - intensive GUI operations in real scenarios

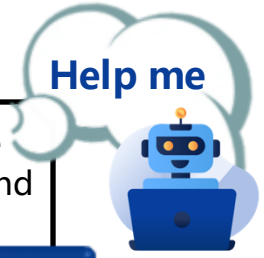
Task: Query the Snowflake database IMDB and save the top 20 dramatic movies since 2000 into file top20movies.csv on Desktop.



Spider2-V: Task Sources and Instructions

Data warehousing

Upload this GoogleSheet to the 'census' datasets in BigQuery and name it 'population'.



Data ingestion and integration

I want to transfer data from Faker to the target database Snowflake. Could you help me setup the source?

Data transformation

Separate the logic of model "customers" out into two staged models, "stg_customers" and "stg_orders".

Data analysis and visualization

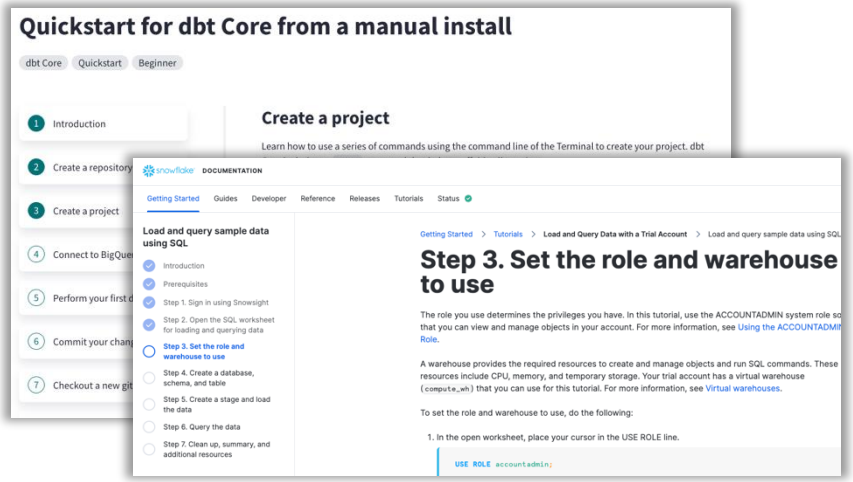
Use dataset "game_sales" to draw a line chart, which should reflect the trend of the average global sales per year.

Data orchestration

I just built a 3-step Dagster pipeline. Could you schedule it to run regularly every hour to keep all assets up to date?

full data pipeline

- **494** real-world tasks
  - collected mainly from official tutorials
  - covering full DS & DEng workflows
  - two versions of task instructions



abstract

- *I have established a connection from Faker to local .csv file. Could you help me change the running schedule? I hope it can be replicated at 6:00 pm every day.*

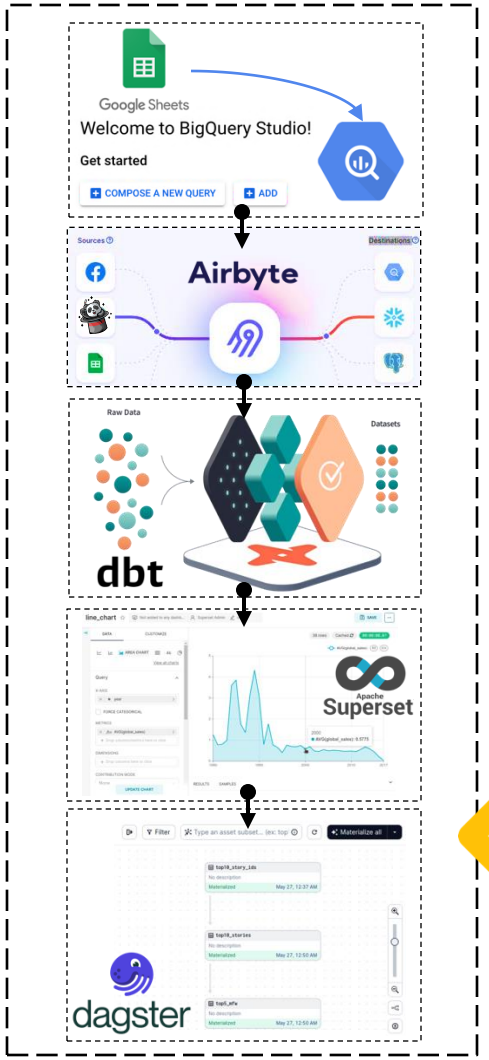
verbose

- *I have established a connection ... at 6:00 pm every day. To finish this task, you can follow these steps:*

1. *Click the connection row ...*
2. *Next, click the "Replication" item ...*
3. *Click the pop-up panel, we will see ...*
4. *In the drop-down options on the right, select the schedule type "Cron" instead of "Scheduled" ...*
5. *Input the value "0 0 18 * * ?" into the cron expression box.*
6. *Finally, click the "Save" button at bottom.*

Spider2-V: Professional Software

- **20 professional enterprise-level** applications
  - even require authentic user accounts
  - crawl documents for RAG framework

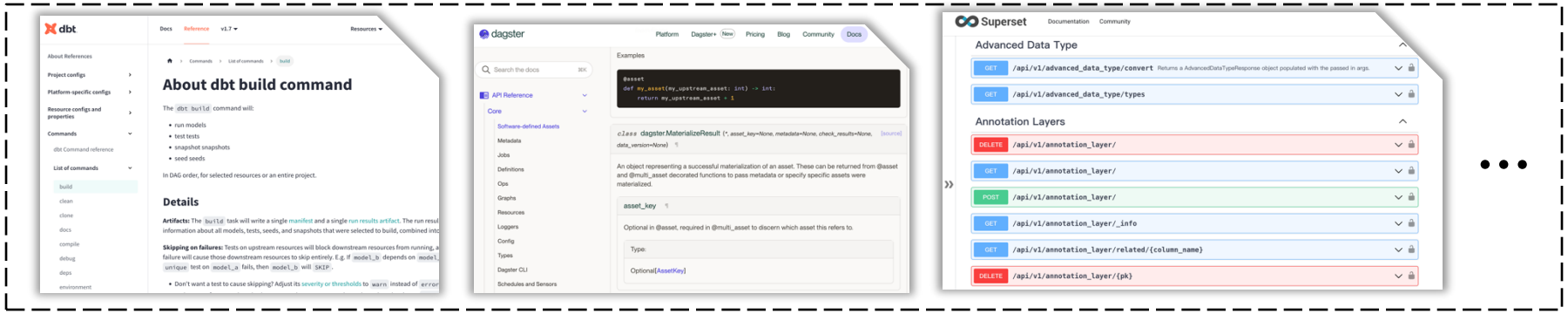


professional tools

```
{
    "account": "https://{xxxxxxxx}.snowflakecomputing.com",
    "user": "USER_NAME",
    "password": "YOUR_PASSWORD"
}
```

JSON

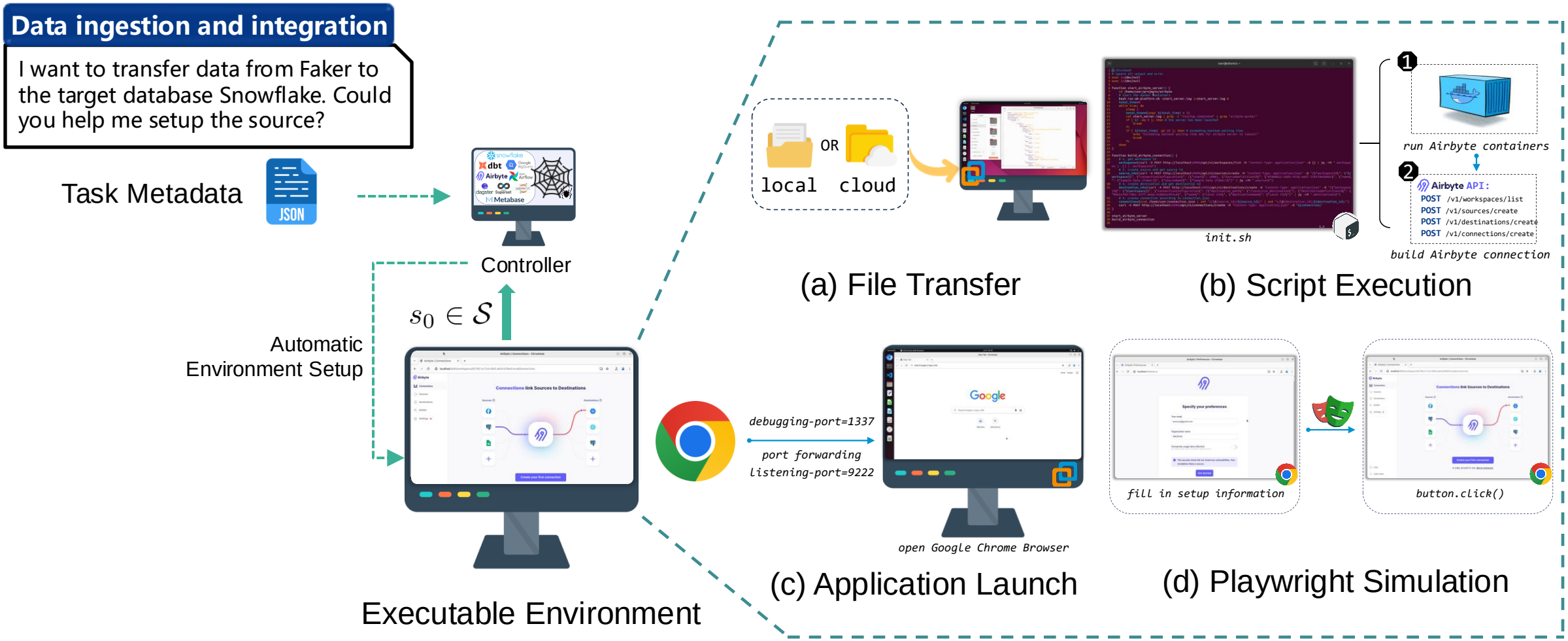
Snowflake account template



documents

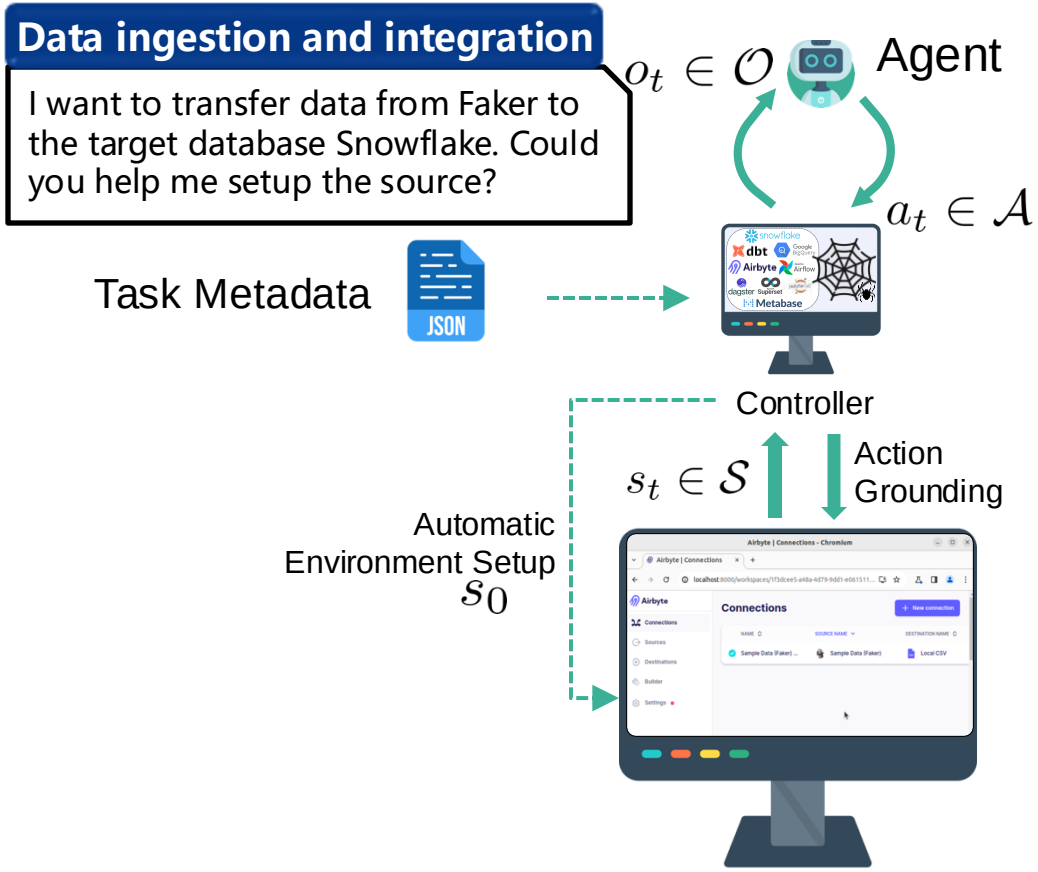
Spider2-V: Environment Setup

- An **interactive executable** computer environment
  - Adapted from OSWorld, based on virtual machines
  - 170 automatic task setup configurations



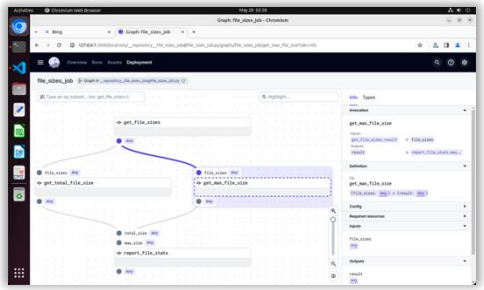
Spider2-V: Action & Observation Space

- Integrate intensive **GUI operations**
  - action space: 1) pyautogui code, or 2) JSON dict
  - observation space: 1) screenshot, and 2) accessibility tree




```
import pyautogui, time
pyautogui.click(100, 100)
pyautogui.typewrite('Hello world!')
pyautogui.press('enter')
time.sleep(0.5)
```

(1) pyautogui code



(1) screenshot

Type	Use case
click	{"type": "click", "x": 12, "y": 46}
move	{"type": "move", "x": 68, "y": 90}
scroll	{"type": "scroll", " clicks": 4}
drag	{"type": "drag", "x": 71, "y": 59}
press	{"type": "press", "key": "enter"}
hotkey	{"type": "hotkey", "keys": ["ctrl", "c"]}
typing	{"type": "typing", "text": "ls -lh"}

(2) JSON dict

```
<desktop-frame name="main" coord="(0,0)" size="(1920,1080)">
  <application name="Chromium" coord="(70, 64)" size="(1442,814)">
    <frame name="Graph: file_sizes_job - Chromium" showing="true" visible="true">
      <push-button name="Minimize" enabled="true">. . .</push-button>
      <push-button name="Maximize" enabled="true">. . .</push-button>
      <push-button name="Close" enabled="true">.. .</push-button>
      <entry name="Address and search bar">http://127.0.0.1:3000/.. .</entry>
      . . . # other elements
    </frame>
  </application>
  <application name="gnome-terminal-server" >
    <frame name="user@ubuntu: ~/fileops-and-jobs/" coord="(70,27)" size="(1442,851)">
      <filler name="" coord="(70, 64)" size="(1442,814)">
        <menu name="File" coord="(70,64)" size="(40,25)" selectable="true">...</menu>
        <menu name="Edit" coord="(110,64)" size="(43,25)" selectable="true">...</menu>
        <menu name="View" coord="(153,64)" size="(49,25)" selectable="true">...</menu>
        <menu name="Help" coord="(337,64)" size="(47,25)" selectable="true">...</menu>
        . . . # other elements
      </filler>
    </frame>
  </application>
  . . . # other application windows
</desktop-frame>
```

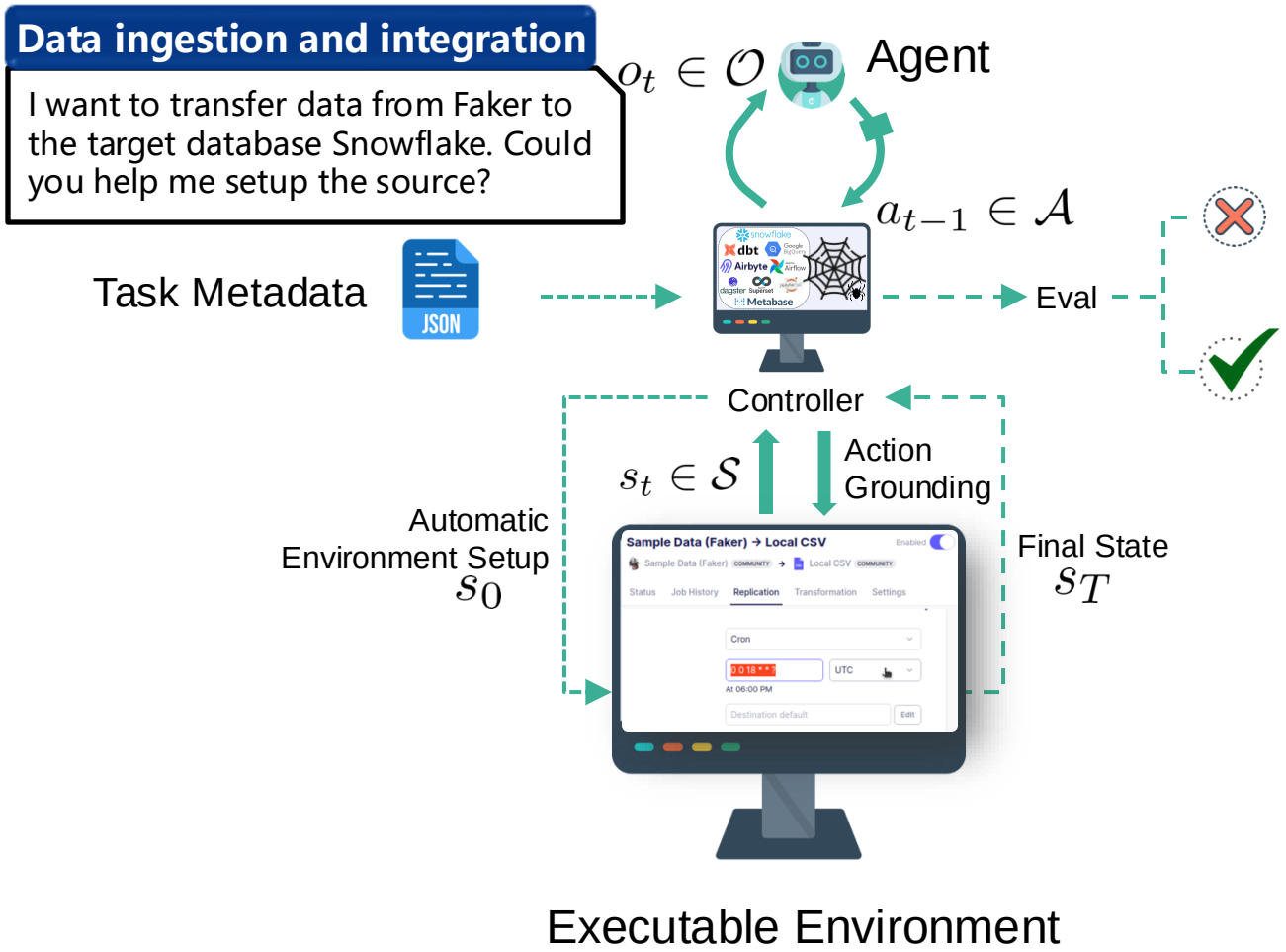
(2) accessibility tree

Action Space $a_t \in \mathcal{A}$

Observation Space $o_t \in \mathcal{O}$

Spider2-V: Execution-based Evaluation

- **Task-specific** evaluation scripts
  - 151 customized metrics/functions in total



Airbyte API:

POST /v1/sources/list

```
curl -X POST \
http://localhost:8000/v1/sources/list \
    -d '{"workspaceId": "xxx-xxx-xxx"}'
```

```
{
    "workspaceId": "xxx-xxx-xxx",
    "connectionConfiguration": {
        "count": 1000
    },
    "sourceDefinitionId": "xxx-xxx-xxx",
    "name": "data transfer source",
    "sourceName": "Faker"
}
```

JSON

information validation

Experiments

Notice: t = temperature, top-p = top-p cutoff, len = max context length, a11ytree = accessibility tree

Rank	Model	Details	Score
1 Jun 3, 2024	GPT-4V (1106) *OpenAI* OpenAI, '23	SoM + EF + RAG t=1.0, top-p=0.9 len = 128k	14.0
2 Jun 2, 2024	GPT-4o (0513) *OpenAI* OpenAI, '24	SoM + EF + RAG t=1.0, top-p=0.9 len = 128k	13.8
3 Jun 5, 2024	Gemini-Pro-1.5 *Google* Gemini Team, Google, '24	SoM + EF + RAG t=1.0, top-p=0.9 len = 128k	9.1
4 June 6, 2024	Claude-3-Opus *AnthropicAI* Anthropic, '24	SoM + EF + RAG t=1.0, top-p=0.9 len = 200k	8.1
5 June 6, 2024	Llama-3-70B *Meta* Meta Llama, Meta, '24	a11ytree + EF + RAG t=1.0, top-p=0.9 len = 32k	2.0
6 June 6, 2024	Mixtral-8x7B *MistralAI* Jiang et al., '24	a11ytree + EF + RAG t=1.0, top-p=0.9 len = 32k	0.8
7 June 6, 2024	Qwen-Max *Qwen* Qwen Team, '24	a11ytree + EF + RAG t=1.0, top-p=0.9 len = 32k	0.6

Table 5: Ablation study on action space, observation types and 3 methods.

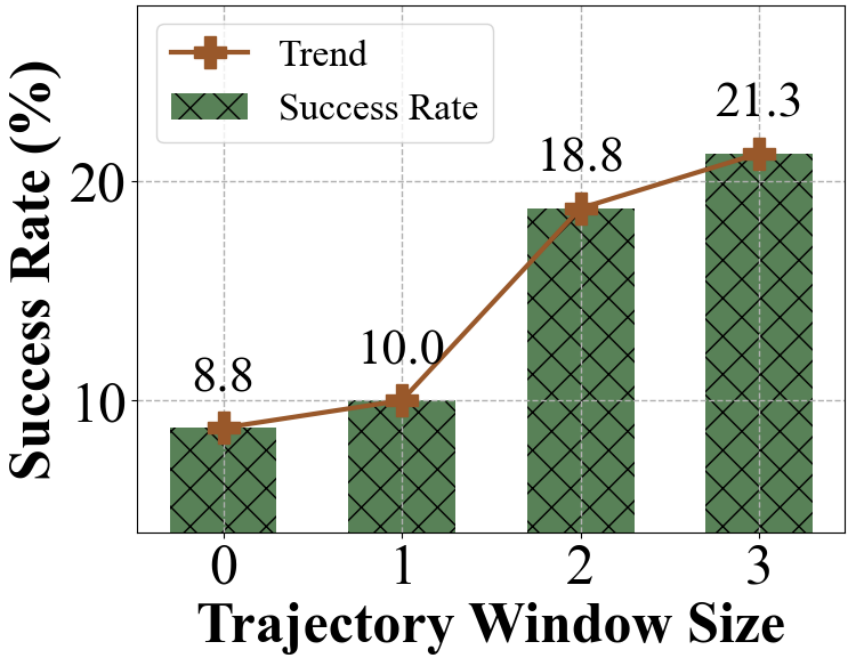
Action Space	Observation Types	SR (%)
JSON dict	screenshot	4.2
pyautogui	screenshot	4.2
JSON dict	a11ytree	10.5
pyautogui	a11ytree	**12.6**
pyautogui	screenshot+a11ytree	11.4
pyautogui	w/ Set-of-Mark	15.6
pyautogui	w/ exec. feedback	13.6
pyautogui	w/ retrieval aug.	14.4
pyautogui	w/ all tricks	**16.3**

- SOTA closed-source models only achieve **14%** success rate, open-source models can hardly solve tasks
- **Action: pyautogui code** > JSON dict ; **Observation: accessibility tree** >> screenshot
- **Set-of-Mark**, **execution feedback**, and **RAG** all contribute to final success

Ablation Study

Table 4: Success rate of GPT-4o on different task partitions.

Task Splits	SR (%)
Easy (# steps ≤ 5)	**38.8**
Medium (# steps $5 \sim 15$)	9.7
Hard (# steps > 15)	1.2
w/o authentic user account	**15.6**
w/ authentic user account	10.6
Abstract instruction	11.3
Verbose instruction	**16.2**



- Tasks with more inherent steps, w/o step-by-step guides, and involving user accounts are **more difficult**
- With the increase of history trajectory window size, performances **improve stably**

Conclusion

- **Spider2-V** is a
  - multi-modal agent benchmark
  - *w.r.t.* data science and engineering
- 494 tasks across **full data workflows**
- 20 **professional enterprise** software
- Integrated CLI and **GUI** actions
- Interactive **executable** environment
- Document warehouse for retrieval

Limitations

- Annotation expensive and not scalable
- Unable to handle time-sensitive tasks
- Long prompt and inefficiency
- Still very poor performances

https://spider2-v.github.io/

Thanks!

https://github.com/xlang-ai/Spider2-V