

Midscene

Midscene, a Chrome extension, was evaluated as part of our testing of an autonomous web agent for extracting and structuring financial data from a bank's website, leveraging Hugging Face Endpoints. To clarify, Midscene is a web-based agent; that we configured to operate with the UI-TARS model, particularly 7-B SFT. This evaluation drew on the previous cohort's approach to agent structuring, though Midscene was not our initial choice, it was one of several options assessed to determine its suitability. The aim was to analyze its ability to streamline financial data collection, utilizing a vision-language model (VLM) to navigate complex webpage layouts and retrieve essential information efficiently.

Technical Implementation

We configured Midscene via PowerShell with the following parameters:

```
OPENAI_BASE_URL=""
```

```
OPENAI_API_KEY=""
```

```
MIDSCENE_USE_VLM_UI_TARS=1
```

```
MIDSCENE_OPENAI_INIT_CONFIG_JSON='{"defaultheaders": {"Authorization":  
""}}'
```

```
MIDSCENE_MODEL_NAME="tgi"
```

Initial token allocation:

- Input tokens: ~40,865
- Maximum new tokens: 2,048
- Total: 42,913 (exceeding the permitted limit of 32,768)


Sample JSON input

```
{  
  "product"  "Dell Laptop"  
  "price"    "$999"  
}
```

BBC Extraction Test (Unsuccessful)

Task - Extract the headline from the BBC homepage.

- **Page content:** ~4,000 tokens
- **Extracted output:** ~500 tokens
- **Outcome:** The total token count exceeded 32,768, resulting in a "Planning422 (Token Limit Exceeded)" error (output below).

 BBC_Midscene_Extension.mkv

Reduction attempts included:

1. `max_input_tokens = 4000, max_new_tokens = 1000` – Unsuccessful
2. `max_input_tokens = 2000, max_new_tokens = 1000` – Unsuccessful
3. `max_input_tokens = 1000, max_new_tokens = 500` – Unsuccessful

Each effort was thwarted by excessive token usage, significantly limiting Midscene's effectiveness.

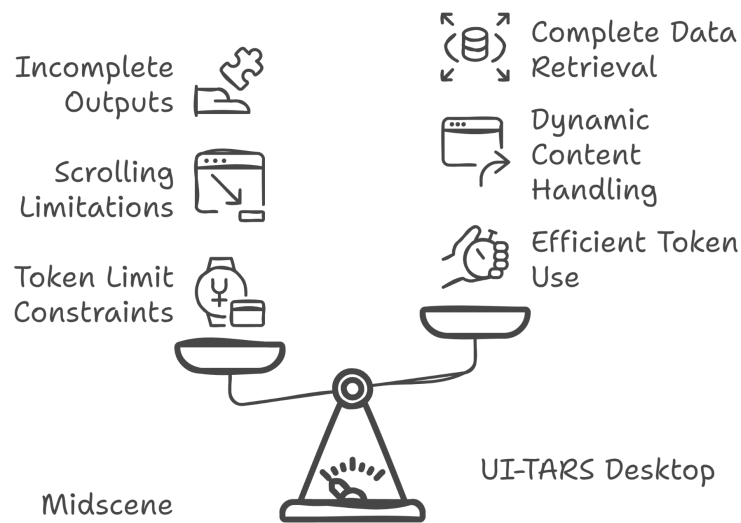
Comparison and Transition to UI-TARS Desktop

Our task was to assess Midscene's performance, specifically its speed, resource efficiency, and overall capability, against the UI-TARS Desktop agent.

Midscene, operating with the UI-TARS model, showed potential for structured data extraction but faced notable challenges:

- **Token Limit Constraints** - Despite adjustments, it consistently surpassed OpenAI's token threshold, hindering larger extractions.
- **Scrolling Limitations** - It struggled with dynamic web page elements, missing content that required scrolling.
- **Incomplete Outputs** - Token restrictions often led to truncated or partial data retrieval.

These shortcomings prompted a shift to the UI-TARS Desktop agent, also utilizing the UI-TARS model, which outperformed Midscene in key areas, as shown in the flowchart below,



Comparing Midscene and UI-TARS Desktop Performance

Conclusion

Midscene, initially tested as a Chrome extension agent paired with the UI-TARS model, provided a practical baseline informed by the previous cohort's efforts to handle web-based tasks.

However, its limitations quickly surfaced, excessive token consumption bogged down processing, and its struggles with dynamic site interactions, such as handling API rate limits (429 errors) and larger webpages, hindered its reliability for real-world financial scraping. These shortcomings prompted a pivot to the UI-TARS Desktop agent, which utilized the same UI-TARS model (primarily 7B DPO) but offered a standalone GUI approach.

This shift proved advantageous: UI-TARS Desktop demonstrated superior navigation through complex web interfaces, greater efficiency in processing multimodal inputs like screenshots, and enhanced reliability in execution-based tasks, such as extracting loan details from HDFC Bank's site. While not without flaws, namely its resource intensity and occasional failure to complete secondary tasks like saving outputs, it outshone Midscene as a more excellent tool for financial data extraction aligning better with our need for practical, dependable performance.