Reproducibility Brief Report

1. Project summary + what I tried to reproduce

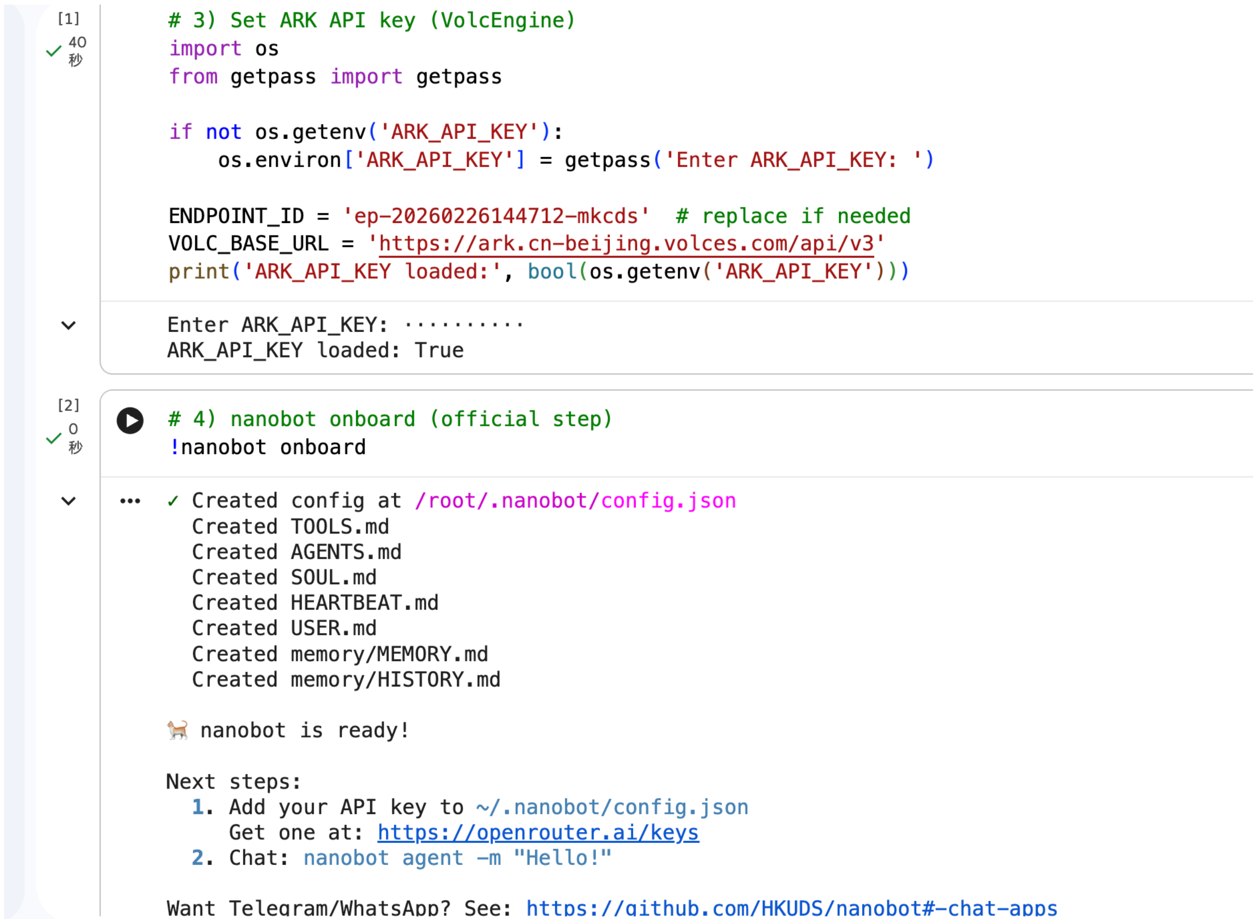
I selected **HKUDS/nanobot** and deployed it in Google Colab with one API key (VolcEngine DeepSeek endpoint). I reproduced a small evaluation workflow and compared two settings:

- baseline: single-pass answering
- modified: self-review style prompt/workflow (same model, same API key)

The goal was to test whether a small prompt/workflow adjustment improves measurable performance.

2. Setup notes (env, data, keys, compute)

- Environment: Google Colab (CPU runtime)
- Main packages: nanobot-ai, openai, pandas, matplotlib, seaborn, tqdm
- API:
  - Base URL: https://ark.cn-beijing.volces.com/api/v3
  - Key: ARK_API_KEY (environment variable only)
  - Model endpoint: VolcEngine DeepSeek endpoint
- Evaluation data:
  - 12 finance tasks per trial
  - categories: terminology, risk, portfolio, calculation, classification
- Compute: 2 trials per setting (n=24 each run)
-



```
# 3) Set ARK API key (VolcEngine)
import os
from getpass import getpass

if not os.getenv('ARK_API_KEY'):
    os.environ['ARK_API_KEY'] = getpass('Enter ARK_API_KEY: ')

ENDPOINT_ID = 'ep-20260226144712-mkcds'  # replace if needed
VOLC_BASE_URL = 'https://ark.cn-beijing.volces.com/api/v3'
print('ARK_API_KEY loaded:', bool(os.getenv('ARK_API_KEY')))
```

```
Enter ARK_API_KEY: ··········
ARK_API_KEY loaded: True
```

```
# 4) nanobot onboard (official step)
!nanobot onboard
```

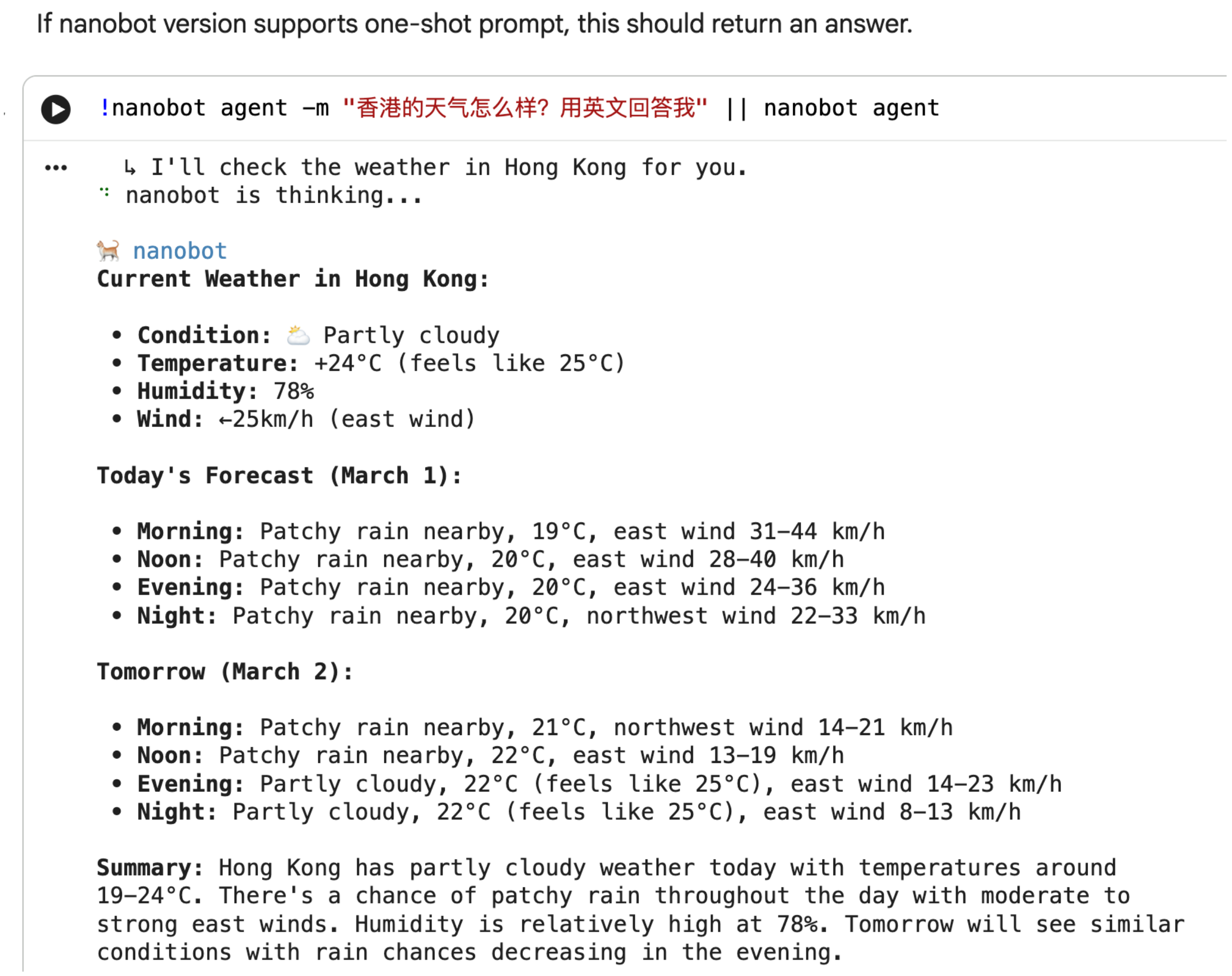
```
✓ Created config at /root/.nanobot/config.json
  Created TOOLS.md
  Created AGENTS.md
  Created SOUL.md
  Created HEARTBEAT.md
  Created USER.md
  Created memory/MEMORY.md
  Created memory/HISTORY.md

🐈 nanobot is ready!

Next steps:
  1. Add your API key to ~/.nanobot/config.json
     Get one at: https://openrouter.ai/keys
  2. Chat: nanobot agent -m "Hello!"

Want Telegram/WhatsApp? See: https://github.com/HKUDS/nanobot#-chat-apps
```

```
Saved: /root/.nanobot/config.json
{
  "agents": {
    "defaults": {
      "workspace": "~/.nanobot/workspace",
      "model": "ep-20260226144712-mkcds",
      "provider": "volcengine",
      "maxTokens": 8192,
      "temperature": 0.1,
      "maxToolIterations": 40,
      "memoryWindow": 100,
      "reasoningEffort": null
    }
  },
  "channels": {
    "sendProgress": true,
    "sendToolHints": false,
    "whatsapp": {
      "enabled": false,
      "bridgeUrl": "ws://localhost:3001",
      "bridgeToken": "",
      "allowFrom": []
    },
    "telegram": {
      "enabled": false,
      "token": "",
      "allowFrom": [],
      "proxy": null,
      "replyToMessage": false
    },
    "discord": {
      "enabled": false,
      "token": "",
      "allowFrom": [],
      "gatewayUrl": "wss://gateway.discord.gg/?v=10&encoding=json"
```



If nanobot version supports one-shot prompt, this should return an answer.

```
!nanobot agent -m "香港的天气怎么样？用英文回答我" || nanobot agent
```

```
  ↳ I'll check the weather in Hong Kong for you.
⠋ nanobot is thinking...

🐈 nanobot
Current Weather in Hong Kong:

 • Condition: ⛅ Partly cloudy
 • Temperature: +24°C (feels like 25°C)
 • Humidity: 78%
 • Wind: ←25km/h (east wind)

Today's Forecast (March 1):

 • Morning: Patchy rain nearby, 19°C, east wind 31-44 km/h
 • Noon: Patchy rain nearby, 20°C, east wind 28-40 km/h
 • Evening: Patchy rain nearby, 20°C, east wind 24-36 km/h
 • Night: Patchy rain nearby, 20°C, northwest wind 22-33 km/h

Tomorrow (March 2):

 • Morning: Patchy rain nearby, 21°C, northwest wind 14-21 km/h
 • Noon: Patchy rain nearby, 22°C, east wind 13-19 km/h
 • Evening: Partly cloudy, 22°C (feels like 25°C), east wind 14-23 km/h
 • Night: Partly cloudy, 22°C (feels like 25°C), east wind 8-13 km/h

Summary: Hong Kong has partly cloudy weather today with temperatures around
19-24°C. There's a chance of patchy rain throughout the day with moderate to
strong east winds. Humidity is relatively high at 78%. Tomorrow will see similar
conditions with rain chances decreasing in the evening.
```

3. Reproduction target(s) + metric definition

Target

Reproduce a stable task-level evaluation pipeline for nanobot usage under my available endpoint, then compare baseline vs modified behavior.

Metrics

- success_rate: non-empty response ratio
- avg_score: task score average (0~1)
- std_score: score variability
- avg_format_ok: format compliance ratio
- avg_latency: average response latency (seconds)

4. Results: my numbers vs reported numbers

The repo does not provide a directly matching official benchmark table for this exact endpoint/task set. So I use baseline as reference and compare modified results.

Overall summary

run_tag	n	success_rate	avg_score	std_score	avg_format_ok
baseline	24	1.0	0.854167	0.345127	1.0
modified	24	1.0	0.812500	0.384835	1.0

Category summary

run_tag	category	avg_score	avg_format_ok
baseline	calculation	0.875	1.0
baseline	classification	1.000	1.0
baseline	portfolio	1.000	1.0
baseline	risk	0.750	1.0
baseline	terminology	0.500	1.0
modified	calculation	1.000	1.0
modified	classification	1.000	1.0
modified	portfolio	0.750	1.0
modified	risk	0.000	1.0
modified	terminology	0.500	1.0

Error sample is empty (no runtime failures).

5. Your modification + results after modification

Modification

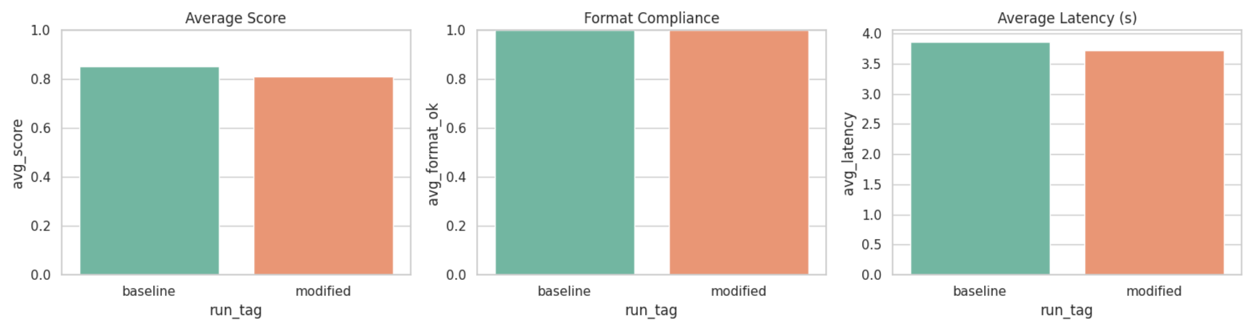
I changed only the prompting/workflow strategy (single-pass vs self-review style), while keeping model/API environment the same.

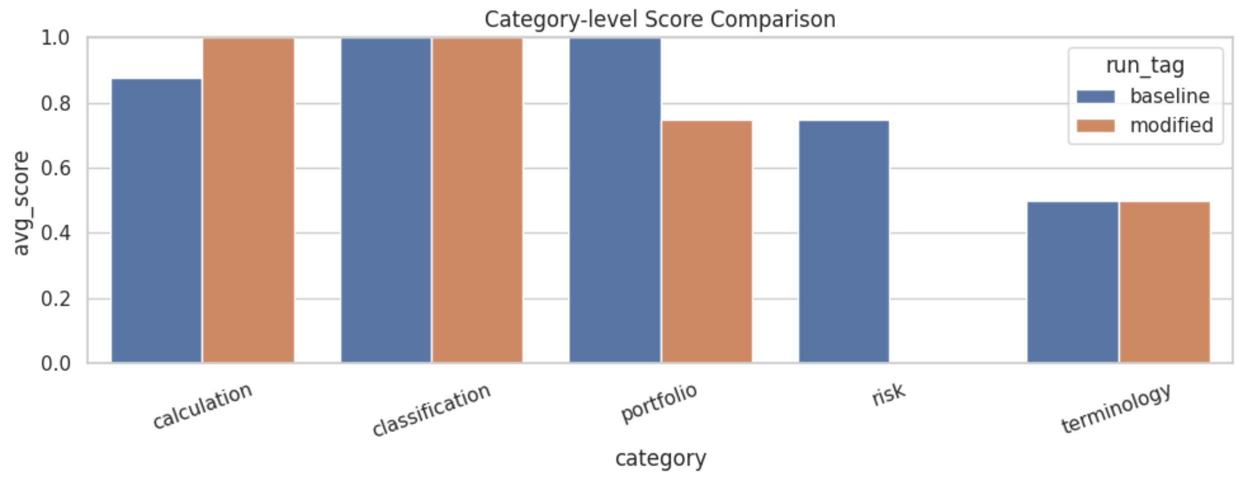
Impact

- avg_score decreased from **0.8542** to **0.8125**
- avg_latency improved from **3.86s** to **3.73s**
- success_rate and avg_format_ok both stayed at **1.0**

Interpretation: modified setting was slightly faster but less accurate overall. It improved calculation tasks but performed worse on risk/portfolio understanding.

Maybe these problems are too simple for the model and it does not need self-review to generate the correct answer. Moreover, I use inappropriate scoring method to evaluate model output, defining the correctness of the answers simply by comparing keywords.





6. Debug diary: main blockers + how I resolved them

- Blocker 1: VolcEngine endpoint did not support web_search tool.

  Fix: removed tool dependency and used pure OpenAI-compatible calls only.
- Blocker 2: Early version had response parsing mismatch in modified run.

  Fix: added robust parsing paths and validation; final run had no errors. (output_text -> output.content.text -> model_dump fallback).

7. Conclusions: what is reproducible, what isn't, and why

Reproducible

- Nanobot deployment on Colab with one API key
- Stable evaluation pipeline

- Quantitative comparison before/after a controlled modification

Not fully reproducible

- Direct comparison to an official published benchmark number for the same endpoint/tasks is not available.

Why

- Endpoint capability differences and no exact official target table for this configuration.