

GAIA项目实战指南

1. 项目概述

1.1 什么是GAIA

GAIA (General AI Assistant) 是由 Hugging Face、Meta AI 等机构联合推出的下一代 **AI 助手基准测试**。与传统只关注文本的测试不同，GAIA 旨在评估 AI 系统在**真实世界场景**中解决复杂问题的能力，被认为是目前最具挑战性的 Agent 评测之一。

- 真实性**: 任务源自人类日常工作，而非合成数据
- 多模态**: 必须同时理解文本、图像、视频、音频和各类文档
- 工具依赖**: 无法仅靠 LLM 内部知识解决，必须熟练使用搜索、浏览器、代码解释器等外部工具
- 复杂推理**: 需要多步骤的规划、推理和纠错能力

1.2 GAIA案例的定位

AWorld 框架中的 GAIA 案例不仅仅是一个演示 Demo，而是一个**生产级 (Production-Ready)** 的通用 AI Agent 参考实现。它基于先进的 **MCP (Model Context Protocol)** 架构设计，展示了如何构建一个能通过 GAIA 高难度测试的 SOTA (State-of-the-Art) 级别 Agent。

它主要展示了：

- MCP 架构的最佳实践**: 如何通过 MCP 协议标准化地集成和管理大量异构工具
- 复杂任务流处理**: 从任务规划、工具调用到结果验证的全自动闭环
- 企业级稳健性**: 包含日志追踪、错误重试、断点续跑等工程化特性
- 多模态融合处理**: 统一处理 PDF/Word/Excel 文档及音视频流的标准化pipeline

1.3 核心特性

✅ **全能多模态支持** 不仅支持常见的 PDF、DOCX、XLSX、PPTX 文档解析，还集成了 ffmpeg 和 libmagic，能够处理和理解图像、音频、视频等流媒体数据，真正实现"全模态"交互。

✅ **强大的 MCP 工具生态 (20+ Tools)** 基于 MCP 协议集成了 16+ 个工具服务 (Server)，提供超过 20 种原子能力：

- 网络交互**: 集成 Playwright 浏览器 (网页操作) 和 Google Search (信息检索)
- 代码执行**: 集成 **E2B 沙箱**，提供安全隔离的 Python/Node.js 代码执行环境
- 专业计算**: 内置 Pandas 数据分析、PubChem 化学搜索、Chess 游戏引擎等专业工具
- 系统控制**: 支持 Terminal 终端操作和文件系统管理

- ✔ **智能推理与规划** 内置 reasoning 推理引擎，支持 Chain-of-Thought (CoT) 思维链，能够将复杂的 GAIA 问题拆解为可执行的子步骤，并根据执行结果动态调整计划。
- ✔ **灵活部署与评估**
 - **双模式运行**：支持交互式 **Web UI** (基于 React) 和自动化 **CLI** 命令行模式
 - **自动化评估**：内置完整的 GAIA 评分系统 (question_scorer)，支持验证集 (Validation Set) 的自动跑测和结果统计
 - **工程化特性**：支持任务黑名单 (Blacklist)、断点续跑 (Resume)、详细的执行日志 (Logging)，方便大规模测试和调试

1.4 技术栈

核心框架：

- AWorld：智能代理编排框架
- MCP (Model Context Protocol)：工具标准化协议
- LangChain：LLM 应用开发框架

编程语言：

- Python 3.12+
- Node.js 22 LTS (WebUI)

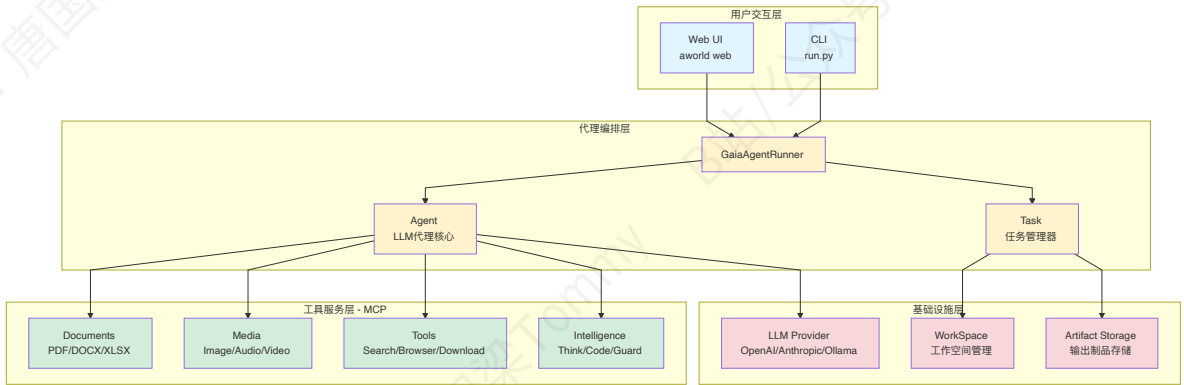
主要依赖：

- playwright：浏览器自动化
- browser-use：智能浏览器代理
- marker-pdf：PDF处理
- anthropic：Claude模型支持
- openai：OpenAI模型支持

2. 架构设计

该项目采用了典型的分层架构，确保了系统的灵活性和可维护性。

2.1 架构分层



1. 用户交互层 (User Interaction Layer)

- **Web UI (aworld web)**: 提供可视化的交互界面, 支持实时流式输出、工具调用展示和结果渲染。
- **CLI (run.py)**: 提供强大的命令行工具, 支持批量评测、断点续跑、黑名单过滤等功能, 适合大规模自动化测试。

2. 代理编排层 (Agent Orchestration Layer)

- **GaiaAgentRunner**: 核心运行器, 负责初始化环境、加载配置、实例化 Agent 以及管理任务生命周期。
- **Agent (AWorld Core)**: 智能体核心, 负责接收任务、规划步骤、调用工具和生成响应。
- **Task**: 封装单个任务的状态和上下文。

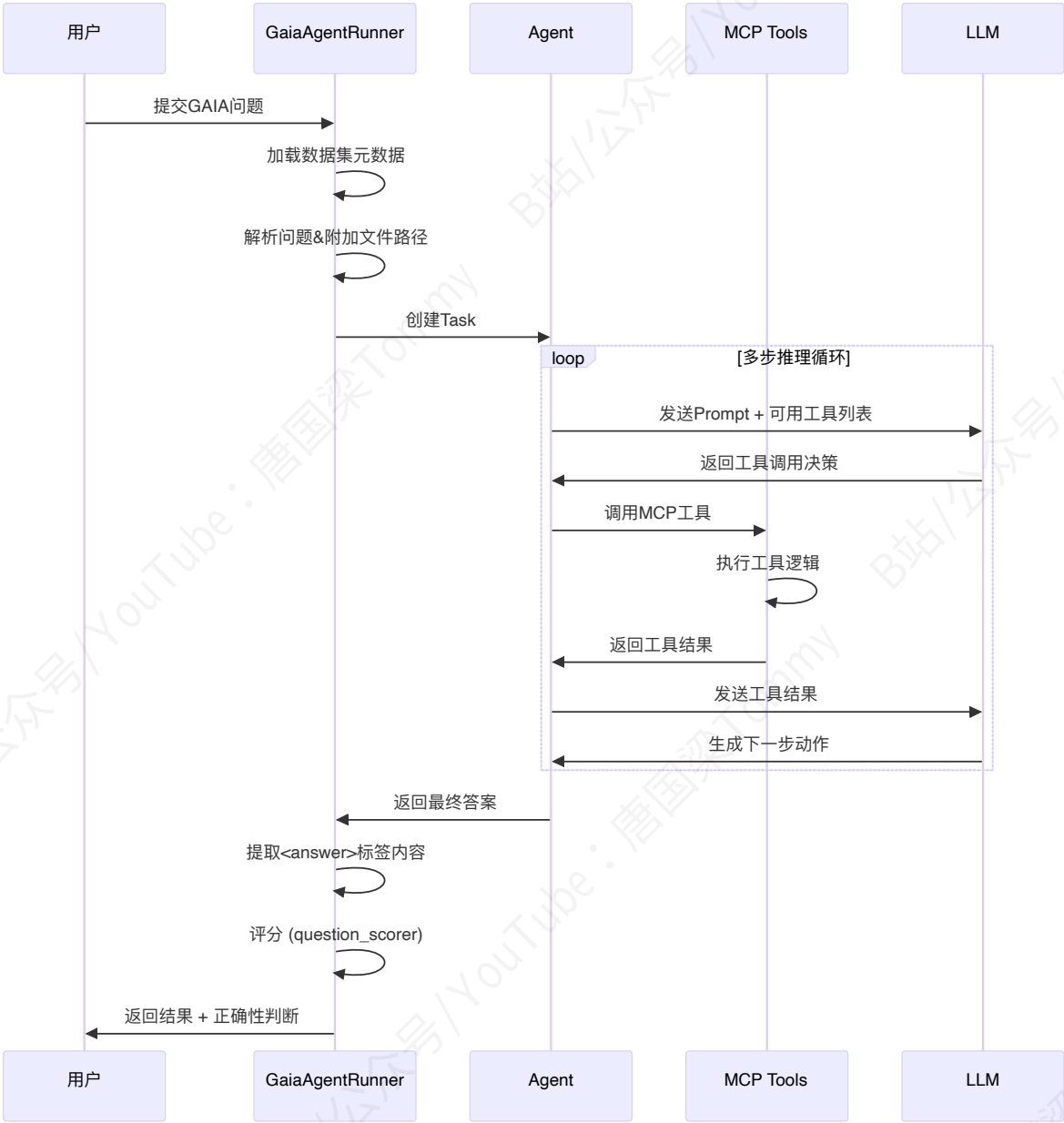
3. 工具服务层 (Tool Service Layer - MCP)

- 项目全面采用 **MCP (Model Context Protocol)** 标准来集成工具。
- 所有工具 (搜索、文档处理、代码执行等) 都封装为独立的 MCP Server。
- 这种设计实现了工具与 Agent 的解耦, 使得添加新工具或替换现有工具变得非常容易, 且支持跨语言实现。

4. 基础设施层 (Infrastructure Layer)

- **Workspace**: 管理任务运行时的文件系统和工作目录。
- **LLM Provider**: 支持MiniMax-M2, KIMI-K2, Qwen3 等多种模型后端。
- **Artifact Storage**: 存储生成的中间产物 (如下载的文件、生成的图表)。

2.2 Agent 工作流



2.3 目录结构

```
examples/gaia/
├── README.md                # 项目说明文档
├── run.py                   # 命令行运行脚本
├── gaia_agent_runner.py     # Agent运行器 (支持Web UI)
├── gaia_agent_server.py    # Agent服务器 (可选)
├── prompt.py               # 系统提示词
├── prompt_w_guard.py        # 带Guard的提示词
├── utils.py                # 工具函数
├── mcp.json                 # MCP配置文件
├── aworld-gaia.yml         # Conda环境配置
├──
├── cmd/                    # 命令行工具
│   └── agent_deploy/
│       └── gaia_agent/
│           ├── agent.py    # Agent定义
│           └── .env.template # 环境变量模板
```


分类	工具名	功能	文件
文档	pdf	PDF解析	documents/pdf.py
	docx	Word文档解析	documents/msdocx.py
	xlsx	Excel表格解析	documents/msxlsx.py
	csv	CSV文件解析	documents/mscsv.py
	pptx	PowerPoint解析	documents/mspptx.py
	txt	文本文件读取	documents/txt.py
媒体	image	图像处理/OCR	media/image.py
	audio	音频转录	media/audio.py
	video	视频帧提取	media/video.py
工具	browser	智能浏览器	tools/browser.py
	search	Google搜索	tools/search.py
	download	文件下载	tools/download.py
	wikipedia	维基百科查询	tools/wiki.py
	youtube	YouTube字幕提取	tools/youtube.py
	wayback	网页历史快照	tools/wayback.py
智能	reasoning	思维链推理	intelligence/think.py
	code	代码执行(E2B)	@e2b/mcp-server
	guard	内容安全检查	intelligence/guard.py
专用	chess	国际象棋	tools/playchess.py
	pubchem	化学数据库	tools/pubchem.py

3.2 多模态文档处理能力

支持处理多种格式:

- 文档: PDF (marker-pdf), DOCX, XLSX, CSV, PPTX, TXT
- 媒体: 图像、音频、视频
- 网页: 浏览器自动化、网页内容提取

3.3 智能浏览器自动化

- Browser-use集成: 基于Playwright的LLM驱动浏览器
- 视觉能力: 启用vision支持
- 智能提示工程: 扩展系统提示处理机器人检测、付费墙、PDF下载等场景
- 结构化输出: 支持markdown/json/text多种格式

3.4 推理增强

- 专用推理模型: 使用KIMI-K2等推理模型处理复杂问题
- 问题分类: 支持数学、代码竞赛、逻辑谜题等场景
- 推理风格: 提供详细/简洁/逐步三种推理模式

3.5 任务编排与控制

- 配置管理: AgentConfig 统一LLM配置
- 会话管理: 通过 session_id 管理会话和工作空间
- 步数限制: max_steps=20 控制任务执行
- 自动评估: 集成GAIA数据集的自动答案评分

4. 环境搭建

4.1 详细安装步骤

步骤1: 获取项目

⚠️ 使用我提供的项目代码文件, 官方项目代码有Bug, 不要使用。

步骤2: 创建Conda环境

```
# 创建并激活环境
conda env create -f examples/gaia/aworld-gaia.yml
conda activate aworld-gaia
```

步骤3: 安装AWorld框架

```
# 进入根目录
cd AWorld/

# 构建Web UI
sh -c "cd aworld/cmd/web/webui && npm install && npm run build"

# 安装AWorld
pip install -e .
```

步骤4: 安装系统依赖

macOS:

```
# 安装Homebrew (如果未安装)
/bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

# 安装系统依赖
brew install libmagic ffmpeg
brew install --cask libreoffice

# 安装Playwright浏览器
pip install playwright
playwright install chromium --with-deps --no-shell
```

Linux (Ubuntu/Debian):

```
# 安装系统依赖
sudo apt-get update
sudo apt-get install -y --no-install-recommends \
    libmagic1 \
    libreoffice \
    ffmpeg

# 安装Playwright浏览器
pip install playwright
playwright install chromium --with-deps
```

步骤5: 下载GAIA数据集


```
# 需要先配置Hugging Face SSH密钥
# 访问: https://huggingface.co/settings/keys
pip install huggingface-hub

# 登录(需要输入密钥tokens)
huggingface-cli login

# 下载数据集
hf download gaia-benchmark/GAIA --repo-type dataset --local-dir
/Users/tangguoliang/Desktop/GAIA
```

步骤6: 配置环境变量

```
# 编辑.env文件, 填写必要的配置
vim gaia/.env
```

4.2 环境变量配置详解

```
# LLM 配置 - 硅基流动
LLM_PROVIDER=openai
LLM_MODEL_NAME=MiniMaxAI/MiniMax-M2
LLM_API_KEY=sk-hkjqqjegn1jrvvnxmkgvuhqppjokjazlcypuxqxihhzhwgyar
LLM_BASE_URL=https://api.siliconflow.cn/v1
LLM_TEMPERATURE=0.0

# 可选配置
BROWSER_HEADLESS=false
BROWSER_WIDTH=1280
BROWSER_HEIGHT=720

# Browser Use 配置 - 用于浏览器自动化任务
# 如果使用 MCP 服务器, 需要单独配置
BROWSERUSE_LLM_BASE_URL=https://api.siliconflow.cn/v1
BROWSERUSE_LLM_MODEL_NAME=MiniMaxAI/MiniMax-M2
BROWSERUSE_LLM_API_KEY=sk-hkjqqjegn1jrvvnxmkgvuhqppjokjazlcypuxqxihhzhwgyar

# =====Path Configurations=====
GAIA_DATASET_PATH="./data/GAIA/2023"
AWORLD_WORKSPACE="/tmp"

# =====MCP Server Configurations=====
# 访问 https://tavily.com 获取 Tavily API Key
TAVILY_API_KEY=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

# [E2B Server](https://e2b.dev/docs/quickstart) 获取 API Key
E2B_API_KEY=e2b_574a97a88989ad5f82f444dc21a1f86d2c802da1
```

```
# Image MCP Server 配置
IMAGE_LLM_MODEL_NAME="Qwen/Qwen3-VL-235B-A22B-Thinking"
IMAGE_LLM_API_KEY="sk-hkjqqjegn1jrvnxmskgvuhqpjokjazlcypuxqxihhzhwgyar"
IMAGE_LLM_BASE_URL="https://api.siliconflow.cn/v1"

# Reasoning MCP Server 配置
THINK_LLM_MODEL_NAME="deepseek-ai/DeepSeek-V3.2-Exp"
THINK_LLM_API_KEY="sk-hkjqqjegn1jrvnxmskgvuhqpjokjazlcypuxqxihhzhwgyar"
THINK_LLM_BASE_URL="https://api.siliconflow.cn/v1"
```

5. 运行方式详解

5.1 Web UI模式

Web UI提供可视化的交互界面，适合调试和演示。

5.1.1 启动步骤

```
# 1. 进入命令目录
cd examples/gaia/cmd

# 2. 启动Web UI
aworld web
```

5.1.2 访问界面

打开浏览器访问: `http://localhost:8000`



Ask or input / use skills



GAIA Agent



5.1.3 使用方式

方式1：运行GAIA任务

输入JSON格式的task_id:

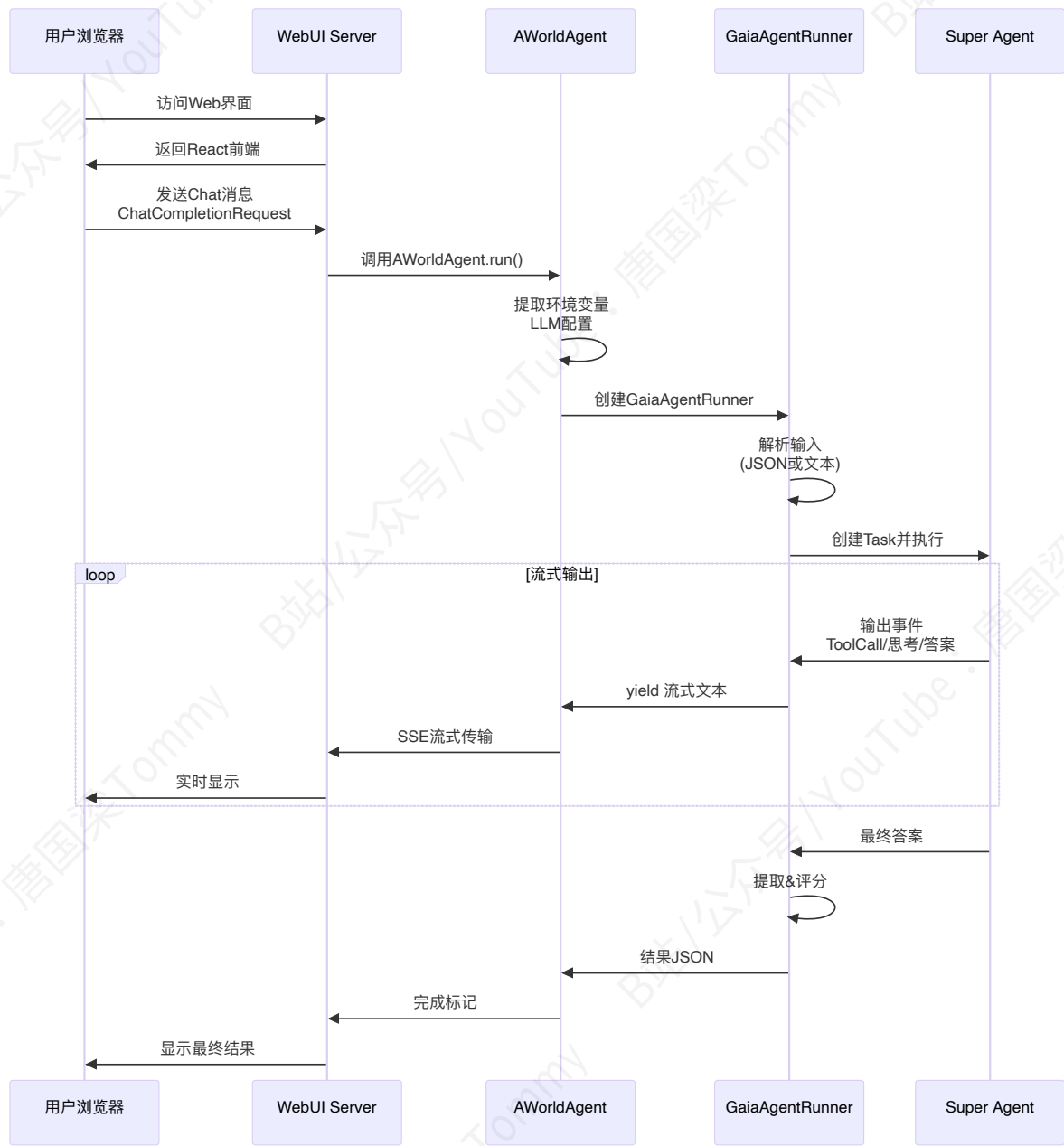
```
{ "task_id": "e1fc63a2-da7a-432f-be78-7c4a95598703" }
```

方式2：自由对话

直接输入问题:

```
帮我查询NVIDIA的最新股价
```

5.1.4 Agent workflow



5.2 命令行模式

命令行模式适合批量处理和自动化。

5.2.1 基础用法

```
# 运行单个问题
cd gaia/

python run.py \
    --split validation \
    --q e1fc63a2-da7a-432f-be78-7c4a95598703
```

```
# 运行一批问题 (索引0-19)
python examples/gaia/run.py \
    --split validation \
    --start 0 \
    --end 2
```

5.2.2 高级用法

1. 跳过已完成的问题

```
python examples/gaia/run.py \
    --start 0 \
    --end 100 \
    --skip
```

2. 使用黑名单

创建黑名单文件 `blacklist.txt`:

```
e1fc63a2-da7a-432f-be78-7c4a95598703
another-task-id
```

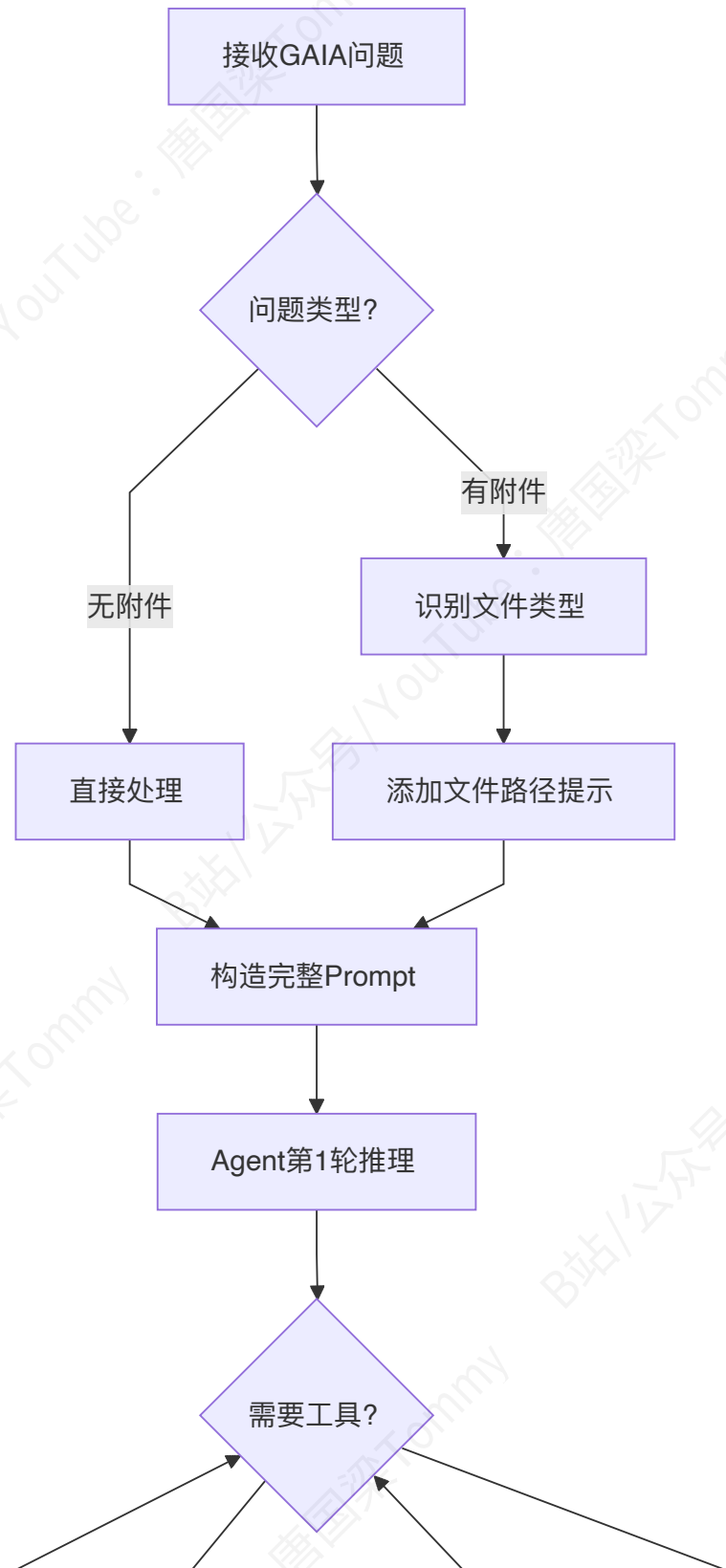
运行时指定:

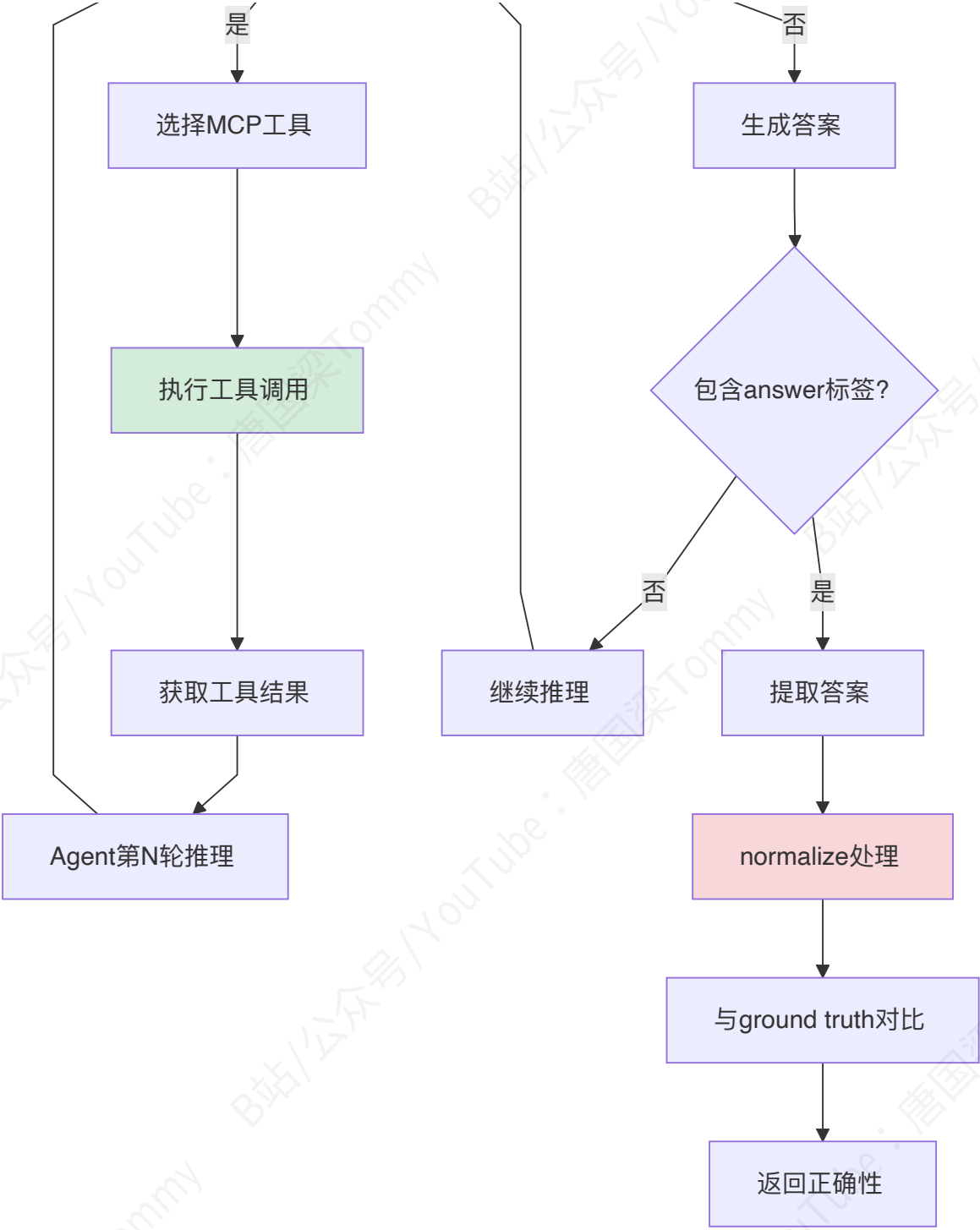
```
python examples/gaia/run.py \
    --start 0 \
    --end 100 \
    --blacklist_file_path blacklist.txt
```

3. 运行测试集

```
python examples/gaia/run.py \  
  --split test \  
  --start 0 \  
  --end 100
```

5.2.3 Agent workflow





AI进阶学习

 B站课堂：

唐国梁Tommy： <https://space.bilibili.com/474347248/pugv>

唐国梁Tommy的精品课： <https://space.bilibili.com/3546748815411393/pugv>

 官方网站： <https://www.tgltommy.com/> (国内访问需科学上网)

 Browse products Home Contact About Me Sign up Log in

Categories

Premium Courses

Public Courses



DeepSeek 系统实战
架构·算法·工程·应用

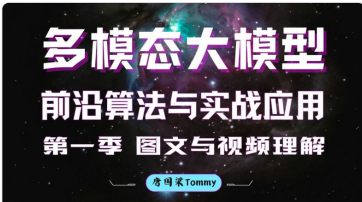
唐国梁Tommy | TGLTommy

DeepSeek系统实战：架构·算法·工程·应用

这是一门面向真实工程落地的大模型系统课程，围绕 DeepSeek 的技术路线，构建从“可控部署与应用搭建”到“核心机制与对齐算法”，再到“系统...”

Course · By 唐国梁Tommy

Learn more



多模态大模型 前沿算法与实战应用
第一季 图文与视频理解

唐国梁Tommy

多模态大模型 前沿算法与实战应用（第一季：图文与视频理解）

本课程是一门面向多模态大模型前沿技术的系列课程，深入探讨了当前多模态大模型的核心技术与最新进展。课程覆盖了从基础概念到高级算法实现...

Course · By 唐国梁Tommy

Learn more



大模型Llama架构 从理论到实战
核心技术与底层原理全解析

唐国梁Tommy

大模型Llama架构：从理论到实战

本课程对 Llama 架构的核心模块进行了深入解析，涵盖算法原理和技术实现。你将学习如何对 Llama模型进行自定义微调、以及本地化模型推...

Course · By 唐国梁Tommy

Learn more



大模型公开课 RAG/Agent/多模态

唐国梁Tommy

大模型公开课（持续更新）

欢迎你加入我的免费公开课~我将在这里将持续分享最新AI前沿技术内容，以简明、实用的方式带你快速了解当下热门的人工智能方法与应用。课程...

Course · By 唐国梁Tommy

Learn more



深入LLM与RAG 原理、实现与应用
大语言模型LLM与检索增强生成RAG理论与实战

唐国梁Tommy

深入LLM与RAG 原理、实现与应用

本课程全面聚焦于检索增强生成 (RAG) 的理论与实践，深入讲解如何结合主流的大语言模型 (LLMs) 来实现智能信息检索和生成。课程从项目环境配...

Course · By 唐国梁Tommy

Learn more

 如果你希望系统掌握大模型核心技术、以及Agent应用开发，推荐你学习我最新上线的精品课程：《DeepSeek 系统实战：架构·算法·工程·应用》

 这是一套从模型微调、部署，到强化学习训练的系统学习路线，课程以企业级落地为目标，你将掌握LLM核心原理、Agentic RAG、MoE/MLA/MTP机制拆解、PPO/GRPO强化学习与工业级 DeepSeek-OCR多模态实战等，想系统掌握并落地这些能力，就从这门课开始。

 你可以在我的B站（唐国梁Tommy的精品课）或个人官网（TGLTommy.com）参与学习：

B站: <https://www.bilibili.com/cheese/play/ss556613313>

官网 (国内访问需科学上网): <https://www.tgltommy.com/p/deepseek>

B站/公众号/YouTube: 唐国梁Tommy

**关注微信公众号
获取更多学习资源**



官方网站: TGLTommy.com(需科学上网)