AI辅助自动化爬取系统使用说明

概述

本系统基于MediaCrawler项目，实现了AI agent辅助的自动化爬取功能，支持微博、B站、知乎三个平台。系统使用大语言模型进行关键词提取和相关性判断，自动筛选与热点事件相关的内容。

主要功能

1. **AI关键词提取**：从热点事件描述中自动提取搜索关键词
2. **AI相关性判断**：对检索结果进行智能相关性判断，过滤不相关内容
3. **多平台支持**：支持微博、B站、知乎三个平台
4. **微博特殊处理**：自动处理微博的search和detail模式，确保获取完整文本
5. **多模态数据**：支持文本、图像、视频数据的爬取

安装和配置

1. 环境要求

- Python 3.9+
- 已安装MediaCrawler项目的所有依赖
- 大语言模型API密钥（OpenAI兼容API）

2. 配置AI Agent

编辑 `config/ai_agent_config.py`：

```
# LLM API配置
LLM_API_KEY = "your-api-key-here"  # 或设置环境变量 OPENAI_API_KEY
LLM_BASE_URL = ""  # 如果使用OpenAI，留空；如果使用其他兼容API，填写base URL
LLM_MODEL = "gpt-4o-mini"  # 模型名称

# 关键词提取配置
MAX_KEYWORDS_PER_EVENT = 5  # 每个事件最多提取的关键词数量

# 相关性判断配置
RELEVANCE_SCORE_THRESHOLD = 0.5  # 相关性评分阈值
ENABLE_RELEVANCE_FILTER = True  # 是否启用相关性过滤
```

3. 配置基础设置

编辑 `config/base_config.py`，确保以下设置：

```
# 无头模式（服务器运行必须）
HEADLESS = True

# 媒体爬取模式（获取图片和视频）
```

```
ENABLE_GET_MEIDAS = True

# 评论爬取模式
ENABLE_GET_COMMENTS = True

# 数据保存方式
SAVE_DATA_OPTION = "json"  # 或 "csv", "sqlite", "db"
```

4. 配置平台特定设置

微博配置 (`config/weibo_config.py`)

```
WEIBO_SEARCH_TYPE = "real_time"  # 搜索类型：default, real_time, popular, video
```

B站配置 (`config/bilibili_config.py`)

```
BILI_SEARCH_MODE = "normal"  # 搜索模式
```

使用方法

方法1：命令行参数

```
python ai_crawler.py "热点事件的具体描述"
```

示例：

```
python ai_crawler.py "某公司发布新产品，引发广泛讨论"
```

方法2：环境变量

```
export EVENT_DESCRIPTION="热点事件的具体描述"
python ai_crawler.py
```

方法3：修改代码

在 `ai_crawler.py` 的 `main()` 函数中直接设置：

```
event_description = "热点事件的具体描述"
```

工作流程

1. **关键词提取**：AI从事件描述中提取搜索关键词
2. **平台搜索**：使用关键词在三个平台进行搜索
3. **相关性判断**：AI判断每个搜索结果是否与事件相关
4. **数据爬取**：
    - 微博：先search获取note_id，判断相关性后，使用detail模式获取全文
    - B站和知乎：直接爬取相关内容的详细信息
5. **数据保存**：保存文本、图像、视频等多模态数据

输出结果

爬取的数据保存在 `data/` 目录下，根据配置的保存方式：

- JSON格式：`data/{platform}/json/`
- CSV格式：`data/{platform}/csv/`
- 数据库：根据配置的数据库类型

注意事项

1. **API密钥**：确保正确配置LLM API密钥，否则AI功能将不可用
2. **登录状态**：首次运行需要登录各平台账号（通过二维码或cookie）
3. **请求频率**：系统已内置请求间隔，避免过于频繁的请求
4. **服务器运行**：确保 `HEADLESS = True`，否则在无图形界面的服务器上可能无法运行
5. **微博全文**：微博的search模式无法获取完整文本，系统会自动使用detail模式获取全文

故障排除

1. AI功能不可用

- 检查API密钥是否正确配置
- 检查网络连接是否正常
- 查看日志中的错误信息

2. 爬取失败

- 检查登录状态是否有效
- 检查平台配置是否正确
- 查看详细错误日志

3. 相关性判断不准确

- 调整 `RELEVANCE_SCORE_THRESHOLD` 阈值
- 使用更强大的模型（如gpt-4）
- 禁用相关性过滤：`ENABLE_RELEVANCE_FILTER = False`

高级配置

自定义相关性判断

可以修改 `ai_agent/llm_agent.py` 中的 `judge_relevance` 方法，自定义判断逻辑。

添加新平台

1. 在 `AICrawlerManager` 类中添加新的爬取方法
2. 实现平台特定的搜索和内容获取逻辑
3. 更新 `crawl_all_platforms` 方法

示例

完整的使用示例：

```
import asyncio
from ai_crawler import AICrawlerManager

async def main():
    event_description = "某热点事件的具体描述，包括时间、地点、人物等关键信息"
    manager = AICrawlerManager(event_description)
    await manager.crawl_all_platforms()

if __name__ == "__main__":
    asyncio.run(main())
```

技术支持

如有问题，请查看：

- MediaCrawler项目文档
- 日志文件中的详细错误信息
- GitHub Issues