

基于Java的低代码主题爬虫的设计与实现

张梦元, 刘莉

(广东白云学院 大数据与计算机学院, 广东 广州 510450)

摘要: 网页数据提取是人工智能与大数据相关课题学习与研究的一项重要内容。为了减轻编写主题网络爬虫程序工作, 该文在主题爬虫的基础原理上, 设计一套通用的Java爬虫程序。程序抽象了下载模块、内容处理模块和结果操作等核心内容, 通过注解或xml配置等低代码方式, 即可实现不同主题内容爬取工作。

关键词: 大数据; Java; 主题爬虫; 低代码

中图分类号: TP311 **文献标识码:** A

文章编号: 1009-3044(2022)30-0033-03

DOI: 10.14004/j.cnki.ckt.2022.1951

开放科学(资源服务)标识码(OSID):



1 引言

大数据与人工智能是国家高度重视的热门研究领域^[1]。中国开展教学研究的高校多, 从事相关科研工作和学习的人员基数大。在从事这些工作和学习中, 收集相关的领域数据是必要的环节。目前市面上有一些开源的数据可供学习使用, 但大部分研究领域数据被相关行业少数公司掌握, 并不对外开放, 给相关研究学习造成了极大的困难。而通过人工的方式从网页提取数据费时费力, 极大影响研究和学习动力。因此在不进行商业盈利和违背法律与爬虫规则的前提下, 开发一些数据爬取工具成为当前数据采集的一个主要途径^[2]。然而, 这类的开发工作有一定的领域门槛, 而且开发程序大多都是一次性的, 对于其他主题的研究无法重复利用。因此, 设计一套低代码的主题爬虫工具, 以期没有爬虫领域知识的需求者也能完成相关主题内容数据的抓取。

2 爬虫的概率和原理

2.1 网络爬虫的概念

网络爬虫也叫蜘蛛程序。因为互联网和蜘蛛网一样, 纵横交错。数据存储在各个网络节点的主机上。网络爬虫就如同蜘蛛一样, 在各个主机节点游走, 获取需要的数据信息。目前将其分为四类, 分别是通用网络爬虫, 聚焦网络爬虫, 增量式网络爬虫, 深层网络爬虫^[3]。本文研究的主题爬虫属于聚焦爬虫这一类, 主要是爬取一些预定的网页数据。由于只访问一些特定的网页, 对网络资源的影响较小。

2.2 主题爬虫的主要原理

主题网络爬虫是利用程序代替浏览器向服务器发送一些特定的地址请求, 从而获取到目标数据。其主要流程是首先下载用户提供起始页面, 获取页面数据和处理相关信息, 然后将网页内链接进行主题相关度计算, 最后根据搜索策略进行下一次链接请求与数据获取^[4-5]。其主要流程如图1所示。

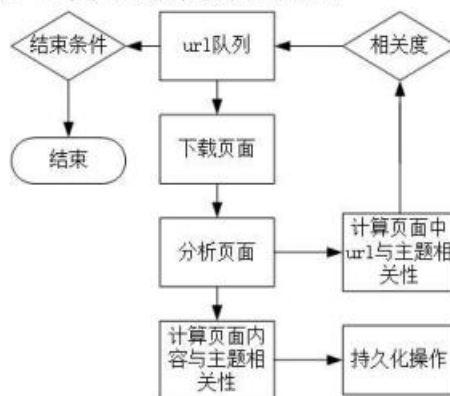


图1 主题爬虫原理

2.3 低代码开发概念

低代码(low code)的核心理念是用少量代码, 或者不写代码来进行系统开发。由此带来的优势有: 降低开发门槛; 减少开发成本; 快速实现业务功能; 系统功能更稳定, 维护性更好。低代码开发分为传统的软件开发、轻量级无代码、企业级低代码等开发形态^[6], 其开发模式的场景适应性从左到右依次增强。

轻量级无代码开发模式以表单驱动, 主要是通过

收稿日期: 2022-08-20

基金项目: 2022年度广东省普通高校重点科研平台, 项目名称: 低代码平台工程技术开发中心(项目编号: 2022GCZX009)

作者简介: 张梦元(1985—), 男, 湖北武汉人, 硕士, 研究方向为软件工程、人工智能; 刘莉(1984—), 女, 湖南衡阳人, 硕士, 研究方向为软件应用、人工智能。

本栏目责任编辑: 谢媛媛

***** 软件技术 ***** 33

预先设定程序规则,由应用开发人员或者业务人员通过图形化界面配置规则或者直接编辑配置规则方式来实现具体业务。

3 爬虫的设计与实现

3.1 爬虫设计

根据爬虫业务需求,给网络爬虫设计下面几个模块:下载器(Download)模块,内容解析器(ContentsResolver)模块,控制器(SpiderController)模块,结果处理器(Dao)模块,整体架构如图2所示。

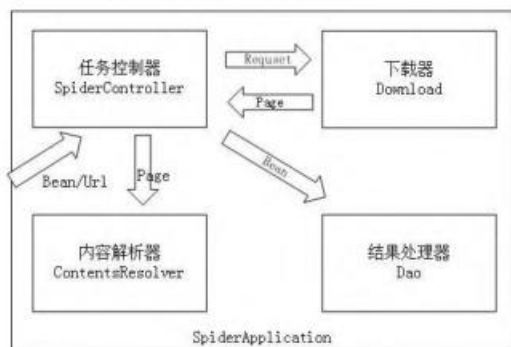


图2 爬虫架构

1) SpiderApplication

SpiderApplication是爬虫程序应用上下文的引导类,通过运行它的run方法启动爬虫应用。其主要流程为加载下载器、内容解析器等实体对象;实例化爬虫控制器,初始化下载监听器、内容处理监听器、结果处理监听器;初始化下载队列,启动下载线程;初始化内容解析队列,启动内容解析线程;初始化结果处理队列,启动结果处理线程。

2) Downloader

数据下载模块根据传递地址信息拉取网络数据并转换成page对象,并将page对象添加到待处理数据队列。下载器是爬虫的通用模块,后续实现其他主题爬虫可以重复利用。处理主要问题构建爬虫与服务间的会话,如登录问题等;确保数据可靠下载,如下载恢复与重试;正确处理文件编码,并正确解析为html文档对象。该模块程序提供完整实现,大多数情况可直接使用,特定要求也支持重载式扩展。其主要流程为首先获取站点配置信息,包括请求头、useragent、用户名、密码,接着建立与站点会话,如需登录则根据配置信息登录,然后下载请求数据,如下载失败重新添加请求队列,并记录重试次数,超过重试次数则丢弃,最后解析数据编码,根据编码转换成html文档保存到page对象。

3) ContentsResolver

内容解析器完成对数据的分析处理。调度器在处理数据队列数据过程中会根据page数据或对应的地址信息请求获取具体解析器对象,由具体解析器实

际处理数据对象。程序定义两种类型的通用解析对象抽象,第一种是链接解析器,即识别并返回后续需处理地址信息,并交由控制器将地址添加到请求队列。第二种是目标数据解析器,提取page中关键信息,后续交由结果处理模块进行相应处理。处理器需在程序初始化过程中注册到应用上下文,程序实现IOC注册框架,实现注册过程可配置化;程序实现解析器可配置式或注解式编写功能,实现低代码式爬虫程序定制。主要流程为用户编写或配置内容处理器,程序启动,获取配置信息,初始化到应用上下文,然后程序调度数据处理队列过程中,根据page信息获取对应处理器,处理器解析数据,返回对应信息,最后控制器根据放回数据类型交由对应模块处理。

4) SpiderController

任务控制器模块实现爬虫地址处理逻辑,下载数据(page)处理逻辑与结果(bean)处理逻辑。通过实现下载回调接口,数据处理回调接口,结果处理回调接口并向下载线程、数据处理线程、结果处理线程添加监听器事件实现信息处理逻辑。记录处理过程中的关键日志。其中下载调度需实现对下载链接信息的去重与持久化工作,持久化的目的是应用程序结束后重启能够恢复前期下载状态。

5) Dao

数据处理模块主要是用来处理提取后的数据,一般用来保存数据,如保存文件和数据库,同时支持对数据进行进一步统计或过滤等功能扩展需求。程序实现数据库与文件配置支持。

3.2 功能实现

3.2.1 下载器模块的实现

Java有许多成熟网络请求框架,程序基于httpClient框架实现。下载器抽象接口定义如下:

```

public interface Downloader {
    public Page download ( String url );
}

```

可通过接口扩展实现不同下载器,通过SpiderApplication的setDownloader进行配置,同时支持xml配置文件。

下载器在页面下载结束后将其转换成Document文档对象。页面解析采用Jsoup框架,该框架支持DOM,CSS方式对页面进行数据提取,后续用户根据需求可简单通过低代码方式扩展相应的内容处理器,实现不同主题爬虫任务。下载器主要代码如下:

```

@com.spider.annotation.Downloader//注解下载器
public class HttpClientDownloader implements
Downloader{
    @Override
    public Page download ( String url ) {

```

```

//1、初始化httpClient对象
HttpClient httpClient = getHttpClient();
//2、构建链接,发送访问请求
httpResponse = httpClient.execute ( getHttpU-
riRequest(),getHttpClientContext());
//3、获取与解析放回数据
//4、解析数据,生成page对象
page = handleResponse ( request , getCharset
(), httpResponse);
//5、回调控制器结果处理逻辑
onSuccess ( request );
return null;
}
}

```

3.2.2 内容处理器的实现

内容处理器是实现不同主题爬虫的关键,程序对其抽象如下:

```

public interface ContentResolver<T> {
    public List<T> doResolve ( Page page );
}

```

接口返回泛型T,如果处理链接解析器,则返回链接字符串。如果处理内容,则可以是用户定义的数据实体类。

该设计目的是实现具体爬虫任务,可以方便进行扩展。同时为了降低编写门槛,程序提供了基于注解或者xml的低代码配置方式。爬虫控制器依据page的url信息,查找对应的内容处理器Resolver。Resolver在应用程序启动时,由SpiderApplication初始化。通过Resolver注解被程序识别,通过type字段设置处理器类型,默认类型为链接处理器。被标识为处理器类需指定SourceUrl注解,控制器将根据value字段来匹配对应内容处理器,value支持正则表达式匹配方式进行模糊匹配。数据解析器需定义返回数据实体方法,数据实体字段通过Select注解标识,Select为抽取规则的封装,基于Jsoup框架实现文档解析,实现Css、Xpath数据提取,同时支持Java正则表达式。Select通过type指定解析类型,通过value设定解析参数。

3.2.3 控制器实现主要逻辑

```

public class SpiderController implements Download-
Listener , ResolveListener {
    @Override
    public void onDownloadSuccess(String path,
Page page) {
        //1、记录下载完成信息
        //2、获取并设置内容处理对象
        //3、添加内容解析队列
    }
}

```

```

@Override
public void onDownloadFail(String path, Result
result) {
    // 1、记录失败信息
    // 2、根据规则丢弃或者重新排队下载
}
@Override
public void onResolveSuccess(DataInfo bean) {
    //1、记录内容解析完成信息
    //2、添加结果处理队列
}
@Override
public void onResolveFail(Page page, Result re-
sult) {
    //增加失败日志
}
}

```

3.2.4 结果处理程序实现

数据处理抽象接口如下:

```

public interface Dao {
    public void process(Object obj);
}

```

由该接口派生文件保存,数据库保存功能。文件保存将内容处理结构对象转换成json字符串保存指定文件夹中,文件名为时间加链接哈希码。数据库保存通过ORM方式。程序定义Table,column注解,指定结果对象存储对应数据库表与列。主要代码逻辑如下:

```

public class OrmDao implements Dao{
    @Override
    public void process(Object obj) {
        Class beanClass = obj.getClass();
        if(beanClass.isAnnotationPresent(Table.class)){
            //1、通过注解获取表明,字段名
            //2、生成对应的sql语句
            //3、通过jdbc保存到数据库
        }
    }
}

```

4 爬虫应用案例

4.1 案例分析与应用

文化传媒行业有大量应用研究案例,比如票房预测、电影推荐等,这些研究内容都是基于电影平台数据^[7-8]。以获取豆瓣网《独行月球》评论信息为例,可基于前文爬虫框架简易实现爬虫程序。

该影片的主页面为<https://movie.douban.com/subject/35183042/>,评论信息页面地址类似如下<https://>

