北京交通大学

基于大数据的智能推荐系统设计与实现

Scrapy爬虫

张迪
dizhang@bjtu.edu.cn

准备工作

- 安装python，pip或者Anaconda
  - Python安装之后添加Path环境变量
  - Pip安装 https://pip.pypa.io/en/stable/installing/
  - Pip安装之后添加环境变量
- Pip install scrapy
  - 需要安装vs buildtools（占用空间很大）
  - https://visualstudio.microsoft.com/downloads/
- 安装IntelliJ IDEA
  - 安装Community版本即可

准备工作

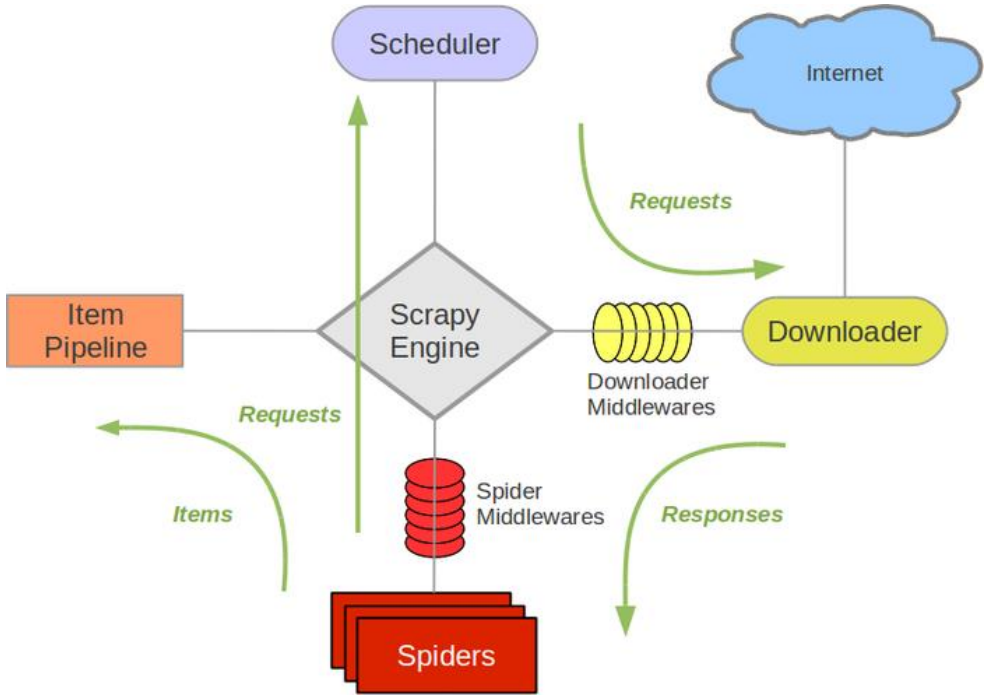
- 配置IntelliJ IDEA
  - 安装python插件
  - Configure→Plugins→Python
- Windows系统
  - Pip install pywin32

小提问

- 什么是爬虫？
  - 是一种按照一定的规则，自动地抓取万维网信息的程序或者脚本
- 哪些地方需要爬虫？
  - 搜索引擎
  - 舆情分析系统
- 你知道哪些爬虫实现框架？
  - Scrapy、PySpider、Crawley、Portia
  - Newspaper、Beautiful Soup、Grab、Cola

Scrapy简介

- Scrapy是什么？
  - 用 Python 实现的一个为了爬取网站数据、提取结构性数据而编写的应用框架
- Scrapy架构
  - Scrapy Engine
  - Scheduler
  - Downloader
  - Spider
  - Item Pipeline



Scrapy简介

- Scrapy Engine（引擎）
  - 负责各个模块之间的通讯，信号、数据传递等
- Scheduler（调度器）
  - 负责接受和调度引擎发送过来的Request请求
- Downloader（下载器）
  - 负责执行下载Requests请求，并将获取到的Responses交还给引擎
- Spider（爬虫）
  - 负责处理所有Responses，从中分析提取数据，获取Item字段需要的数据，并将需要跟进的URL提交给引擎，再次进入Scheduler(调度器)

Scrapy简介

- Item Pipeline（管道）
  - 负责处理Spider中获取到的Item，并进行进行后期的处理（详细分析、过滤、存储等）
- Downloader Middlewares（下载中间件）：可以当作是一个可以自定义扩展下载功能的组件
- Spider Middlewares（Spider中间件）：可以理解为是一个可以自定扩展和操作引擎和Spider中间通信的功能组件

制作Scrapy爬虫

- 1. 新建项目
  - scrapy startproject scrapy_book
- 2. 明确目标
  - 明确想要抓什么（编写item.py）
- 3. 制作爬虫（最重要的一步）
  - 设计爬虫（编写spiders/spider_xpath.py）
- 4. 存储内容
  - 设计管道存储爬取内容（编写piplines.py）

爬虫示例：爬取豆瓣图书

- item.py
  - 图书信息

```python
import scrapy


class ScrapyBookItem(scrapy.Item):
    # 图书id、书名、作者、出版日期、出版社、小图、中图、大图、类别id、类
    isbn = scrapy.Field()
    # 书名
    title = scrapy.Field()
    # 作者
    author = scrapy.Field()
    # 出版日期
    pub_date = scrapy.Field()
    # 出版社
    publisher = scrapy.Field()
    # 小图
    s_img = scrapy.Field()
    # 中图
    m_img = scrapy.Field()
    # 大图
    b_img = scrapy.Field()
    # 类别id
    category_id = scrapy.Field()
    # 类别名称
    category = scrapy.Field()
    # 单价
    price = scrapy.Field()
    # 图书一句话描述
    scrible = scrapy.Field()

    sub_url = scrapy.Field()
    img = scrapy.Field()
    score = scrapy.Field()
    num = scrapy.Field()
```

爬虫示例：爬取豆瓣图书

- spiders
  - 开始请求
  - 解析

```python
# -*- coding: utf-8 -*-
import scrapy

from scrapy_book.items import ScrapyBookItem


class SpiderForXPath(scrapy.Spider):
    name = 'spider_xpath_douban'

    def start_requests(self):
        for a in range(10):
            url = 'https://book.douban.com/top250?start={}'.format(a * 25)
            yield scrapy.Request(url=url, callback=self.parse)


    def parse(self, response):
        items = []

        for book in response.xpath('//*[@id="content"]/div/div[1]/div/table'
            item = ScrapyBookItem()

            title1 = book.xpath("./tr/td[2]/div[1]/a/@title").extract_first(
            title2 = "无" if book.xpath("./tr/td[2]/div[1]/span/text()").extr
                "./tr/td[2]/div[1]/span/text()").extract_first().replace('\n
            item['title'] = title1 + "(" + title2 + ")"
            item['s_img'] = book.xpath("./tr/td[1]/a/img/@src").extract_firs
            item['scrible'] = "无" if book.xpath("./tr/td[2]/p[2]/span/text(
                "./tr/td[2]/p[2]/span/text()").extract_first().replace('\n',
            sub_url = book.xpath("./tr/td[2]/div/a/@href").extract_first().r
            items.append(item)

            # meta={"item":item} 传递item引用SinaItem对象
            yield scrapy.Request(url=sub_url, callback=self.parse_second, me

    def parse_second(self, response):
        item = response.meta["item"]
        item["category_id"] = ""
        item["category"] = ""
        # book = response.xpath('//div[@class="indent"]/div').extract_first(
        # item["author"] = book.xpath("./div[1]/a[1]/text()").extract_first(
```

爬虫示例：爬取豆瓣图书

- piplines.py
  - 输出

```python
import codecs
import json


class ScrapyBookPipeline(object):
    def __init__(self):
        self.file = codecs.open('data.json', 'w', encoding='utf-8')

    def process_item(self, item, spider):
        line = json.dumps(dict(item), ensure_ascii=False) + "\n"
        self.file.write(line)
        return item

    def spider_closed(self, spider):
        self.file.close()
```

爬虫任务

- 制作一个针对其他图书网站的爬虫程序
  - 京东、当当、亚马逊等
- 参考资料
  - scrapy_book.rar
  - 基于Scrapy的爬虫

Thank you!

Q & A