# 豆瓣电影Top250爬虫系统课程设计报告

# 豆瓣电影Top250爬虫系统课程设计报告

## 1. 选题描述

### 1.1 项目背景

豆瓣网作为中国最大的文化娱乐评分平台之一，其电影Top250榜单集合了大量高质量电影信息和用户评论。本项目旨在开发一个自动化爬虫系统，抓取豆瓣电影Top250榜单上的电影信息及其评论数据，建立本地数据库进行存储和分析。

### 1.2 选题意义

* 实践Web爬虫技术，了解网页数据抓取的基本流程和技术要点
* 学习数据解析和处理方法，提升Python编程能力
* 掌握数据库设计和操作技能
* 为后续的电影推荐系统或数据分析项目提供数据基础

### 1.3 技术路线

本项目基于Python语言开发，使用requests库进行HTTP请求，BeautifulSoup库进行HTML解析，PyMySQL进行数据库操作，实现从豆瓣网自动化抓取数据并存储到MySQL数据库的功能。

## 2. 方案设计

### 2.1 系统架构

本系统采用简单的三层架构设计：

* 数据抓取层：负责从豆瓣网获取原始HTML数据
* 数据处理层：解析HTML内容，提取所需信息
* 数据存储层：将处理后的数据保存到MySQL数据库

### 2.2 技术选型

* 开发语言：Python 3.x
* 网络请求：Requests库
* HTML解析：BeautifulSoup4
* 数据库：MySQL
* 数据库连接：PyMySQL

### 2.3 数据库设计

#### 2.3.1 movies\_list表

存储电影基本信息

id (主键) - 自增ID

top25No - 排名

title - 电影标题

year - 上映年份

director - 导演

scriptwriter - 编剧

lead\_performer - 主演

genre - 类型

produced\_country\_or\_region - 制片国家/地区

language - 语言

initial\_release\_date - 上映日期

runtime - 片长

also\_known\_as - 别名

IMDb - IMDb编号

official\_site - 官方网站

summary - 剧情简介

rating - 评分

nums\_of\_rating\_people - 评分人数

ratings\_on\_weight - 评分占比

rating\_betterthan - 好于同类电影比例

comments\_site - 评论页面链接

#### 2.3.2 comments\_list表

存储电影评论信息

id (主键) - 自增ID

movie\_id (外键) - 关联movies\_list表的id

user - 评论用户

star - 用户评分

time - 评论时间

useful - 有用数量

content - 评论内容

## 3. 代码模块功能划分与描述

### 3.1 网络请求模块

该模块负责发送HTTP请求并获取网页内容，主要通过`fromurl2soup`函数实现。该函数执行以下操作：

* 设置随机Cookie和User-Agent模拟浏览器行为
* 发送GET请求获取页面内容
* 使用BeautifulSoup解析HTML

def fromurl2soup(url):

# 代码实现省略...

# 功能：发送HTTP请求并将返回的HTML转换为BeautifulSoup对象

### 3.2 电影列表抓取模块

实现通过`crawl\_top25`函数抓取豆瓣Top250的电影链接列表。主要功能：

* 定位页面中的电影列表元素
* 提取每部电影的详情页链接
* 处理可能的异常情况

### 3.3 电影详情解析模块

通过`parse\_movie\_detail`函数解析每部电影的详细信息，包括：

* 电影基本信息（标题、年份、导演等）
* 电影评分和评分分布
* 剧情简介
* 相关链接

该模块设计了通用的信息提取函数`find\_pure\_text\_tag`，实现了对不同类型数据的统一处理。

### 3.4 评论抓取与解析模块

该模块分为两部分：

* `parse\_comments\_site`：抓取电影评论页面，获取评论元素列表
* `parse\_comments\_div`：解析单条评论信息，提取用户、评分、内容等数据

### 3.5 数据库操作模块

负责数据库和表的创建、数据插入等操作：

* `create\_database`：创建MySQL数据库
* `create\_movies\_table`和`create\_comments\_table`：创建电影和评论表
* `insert\_movie\_table`和`insert\_comments\_table`：插入电影和评论数据

## 4. 实现效果

### 4.1 运行效果

截图见文末

### 4.2 数据采集结果

截图见文末

### 4.3 性能分析

* 成功抓取电影数：25部
* 评论数据量：每部电影约30条评论，共约750条评论

### 4.4数据库导入

* 打开命令行终端（Windows的CMD/PowerShell或Mac/Linux的终端）
* 先登录MySQL: mysql -u 用户名 -p
* 创建一个数据库（如果需要）: CREATE DATABASE movies;
* 退出MySQL环境: exit;
* 在命令行中使用以下命令导入SQL文件（指定自定义路径）:
* # Windows系统
* mysql -u 用户名 -p movies < C:\您的自定义路径\movies.sql
* # Mac/Linux系统
* mysql -u 用户名 -p movies < /您的自定义路径/movies.sql

## 5. 问题与解决方案

### 5.1 反爬虫措施应对

豆瓣网有一定的反爬虫机制，本项目通过以下方式解决：

* 随机Cookie值
* 模拟浏览器User-Agent

### 5.2 页面解析挑战

豆瓣页面结构复杂且可能变化，通过以下方式提高解析稳定性：

* 多级定位，先找到大区域再定位具体元素
* 异常处理，防止因单个元素解析失败导致整个程序崩溃
* 灵活解析方法，针对不同数据类型使用不同解析策略

## 6. 总结与展望

### 6.1 项目总结

本项目成功实现了对豆瓣电影Top250的数据抓取和存储，掌握了Web爬虫的基本技术和数据处理方法，为后续的数据分析奠定了基础。

### 6.2 改进方向

* 添加代理IP池，进一步规避反爬措施
* 实现多线程或异步爬取，提高爬取效率
* 增加增量更新功能，定期更新数据
* 开发数据可视化界面，展示分析结果
* 扩展爬取范围，如演员信息、更多评论等

## 7. 参考资料

1. BeautifulSoup官方文档: https://www.crummy.com/software/BeautifulSoup/bs4/doc/

2. Requests官方文档: https://requests.readthedocs.io/

3. PyMySQL文档: https://pymysql.readthedocs.io/

运行效果截图: 

