**数据采集技术项目训练报告**

爬取日本动漫电影top100

院系名称：电气与信息工程学院

专业：数据科学与大数据技术

学号：

班级：18大数据1班(本)

姓名：

指导教师：唐保存

训练时间：2020年6月22日--2020年6月26日

**一、项目选择的背景及意义**

**背景：** 经过一个学期的python学习，学习到了数据采集的技术，通过此次项目，来对所学的知识做一个总结和实践，检验自己的学习成果。

**意义：**动漫对于很多人来说都是一种特别的情怀，不管大人还是小孩都是情有独钟，爬取日本动漫top 100 让更多的人了解和观看经典动漫是一个不错的想法。在经过学习之后自己有了能力去获取这些资讯，使得更多的人省去了百度的时间，是一个十分有意义的事情。

**二、项目总体设计及使用说明**

1. 目标网站：<http://movie.mtime.com/list/1709.html，通过这个网站获取电影排行的id，方便拼接url爬取详情信息。>

2. 详情url：[http://movie.mtime.com/list/1709-1.html，此处的“1”为电影的id](http://movie.mtime.com/list/1709-1.html，此处的\“1\”为电影的id) 通过目标url来获取拼接。

3. 使用BeautifulSoup函数来将网页进行解析，获取目标id与电影的数据详情。

4. 使用说明：

填写下面的header信息后运行即可

榜单页面：http://movie.mtime.com/list/1709.html

(1).(Google Chrome)

打开榜单页面F12点击NetWork里的Doc选项-->刷新页面-->右键1709.html-->copy-->copy as cURL

(2).打开 https://curl.trillworks.com/ 网站工具

(3).粘贴已复制的cURL到左边输入框,直接转换成Python形式的请求

(4).复制header信息并粘贴下面到相应位置

(5).生成的数据保存在当前文件夹下，是一个json文件。

(6). 在本项目中都是配置好的，只需要安装环境后之间运行即可。

ps：反爬虫，模拟成浏览器访问网站，避免请求失败

**三、项目设计体会总结及展望**

现在的时代也是信息的时代，当前数据爆炸式增长，各种数据繁多，网络爬虫作为一种获取数据的基本手段也渐渐的为人们所掌握，用来获取一定的数据，来对自己的业务做一定的辅助分析或者是学者们做科学计算的语料库，但是每个服务器的承载能力是有限的，短时间内发起大量的请求对于目标服务器是一种致命的伤害，我们一定要合理合法的使用网络爬虫获取数据，方便他人就是方便自己。

通过这次爬虫实验，让我知道了如何在非结构化的数据中提取数据并且进行结构化，方便使用。也学到了网络请求的过程，通过各种手段伪装自己的请求，去获取到一些公开的数据，去展示或者用这些数据去研究一些新的技术，比起人工收集数据来说效率不是提高了一点点而是大幅度的减少了人力的成本，让人力去做更多机器做不到的事情，这是技术的发展也是人类社会的进步，计算机的作用越来越大，计算机越来越能代替人工做一些机械性的重复的工作。

爬虫和反爬虫的斗争在爬虫的产生阶段就已经发生了，但是往往都是以爬虫的胜利为结束，成功的突破了反爬虫的限制获取到想要的数据。当然一些比较大型的网站的反爬虫是很厉害的，需要更加高深的爬虫技术才能突破获取到数据，这也是爬虫工程师的由来，双方的技术都在不停的攻防中进步，整体上推动了技术的发展，近些年机器学习的火热也同样应用在了反爬和爬虫的领域中。

经过这次的学习，认识到自身能力的不足以及日后的发展方向。学习是无止境的，技术是日新月异的，只有不停的前进才能跟上社会的脚步，跟上技术的发展，与时代共同进步。

1. **附项目源程序**

**import requests**

**from bs4 import BeautifulSoup**

**import json**

**import re**

**from tqdm import tqdm**

**"""**

**使用方法：**

**填写下面的header信息后运行即可**

**榜单页面：http://movie.mtime.com/list/1709.html**

**1.(Google Chrome)**

**打开榜单页面F12点击NetWork里的Doc选项-->刷新页面-->右键1709.html-->copy-->copy as cURL**

**2.打开 https://curl.trillworks.com/ 网站工具**

**3.粘贴已复制的cURL到左边输入框,直接转换成Python形式的请求**

**4.复制header信息并粘贴下面到相应位置**

**ps：反爬虫，模拟成浏览器访问网站，避免请求失败**

**"""**

**def headers():**

**#headers 为爬虫所必须，伪装成浏览器进行访问，其中user-agent是浏览器的头，标识访问是通过什么浏览器去访问的**

**headers = {**

**'Connection': 'keep-alive',**

**'Cache-Control': 'max-age=0',**

**'DNT': '1',**

**'Upgrade-Insecure-Requests': '1',**

**'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.138 Safari/537.36',**

**'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.9',**

**'Accept-Language': 'zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,zh-TW;q=0.6',**

**}**

**return headers**

**# 获取Top100动漫的id**

**def get\_ids():**

**ids = []**

**# 向网站发起请求，并获取响应对象**

**response\_one = requests.get('http://movie.mtime.com/list/1709.html',**

**headers=headers(), verify=False)**

**#获取返回的内容**

**html\_one = response\_one.text**

**soup\_one = BeautifulSoup(html\_one, 'lxml') # 选择lxml作为解析器**

**id\_info = soup\_one.select('div[class="top\_nlist"]') # 这个属性所在的div标签内是电影排行信息**

**patt = r'<a href="http://movie.mtime.com/(\d+)/' #使用正则表达式来获取对应的数据**

**id\_one = re.findall(patt, str(id\_info), re.S)**

**# 将第一页排行榜的电影id存入ids列表**

**for item in id\_one:**

**ids.append(item)**

**for num in range(2, 11):**

**# 总共10页数据，通过.format(num)替换{}为页数来实现榜单翻页，向网站发起请求**

**response = requests.get('http://movie.mtime.com/list/1709-{}.html'.format(num),**

**headers=headers(), verify=False)**

**html = response.text**

**soup = BeautifulSoup(html, 'lxml')**

**id\_info = soup.select('div[class="top\_nlist"]')**

**patt = r'<a href="http://movie.mtime.com/(\d+)/'**

**id\_one = re.findall(patt, str(id\_info), re.S)**

**for item in id\_one:**

**ids.append(item)**

**print('Top100动漫的id已获取:')**

**print(ids)**

**return ids**

**# 获取动漫的导演和编剧等信息**

**def get\_movie\_info(id):**

**# 通过.format(id)替换{}为id来实现向指定电影详细页面发起请求**

**response = requests.get('http://movie.mtime.com/{}/'.format(id), headers=headers())**

**html = response.text**

**soup = BeautifulSoup(html, 'lxml')**

**info\_name = soup.select('dd[pan="M14\_Movie\_Overview\_BaseInfo"]') # 这个属性所在的dd标签内是电影信息**

**info\_names = re.findall(r'</strong>(.\*?)</a>', str(info\_name), re.S)**

**data = []**

**# 捕获由于页面改动所发生的异常**

**try:**

**director = str(info\_names[0]).split('target="\_blank">')[-1] # 导演**

**playwriter = str(info\_names[1]).split('target="\_blank">')[-1] # 编剧**

**section = str(info\_names[2]).split('target="\_blank">')[-1] # 发行地区**

**company = str(info\_names[3]).split('target="\_blank">')[-1] # 发行公司**

**movie\_info = soup.select('div[class="db\_cover \_\_r\_c\_"]')**

**movie\_names = re.findall(r'title="(.\*?)">', str(movie\_info), re.S) # 提取电影名称片段**

**movie\_name = re.findall(r'\[\'(.\*?)\(', str(movie\_names), re.S) # 提取电影名称**

**movie\_title = movie\_name[0]**

**movie\_times = re.findall(r'(\d+)', str(movie\_names), re.S) # 提取电影发行时间**

**movie\_time = movie\_times[0]**

**#将获取到的信息组合成一个字典**

**info = {**

**'电影名': movie\_title,**

**'导演': director,**

**'编剧': playwriter,**

**'国家地区': section,**

**'发行公司': company,**

**'发行时间': movie\_time**

**}**

**print(info)**

**data.append(info)**

**except IndexError as e:**

**print(e)**

**return data**

**def write2json(lists):**

**with open('./T100数据.json', 'w', encoding='utf-8') as f:**

**# ensure\_ascii=False 不使用用ASCII编码，防止中文乱码**

**json.dump(lists, f, ensure\_ascii=False)**

**print("保存成功")**

**if \_\_name\_\_ == '\_\_main\_\_':**

**#获取id的函数，通过此函数获取top100 的电影的id方便拼接**

**ids = get\_ids()**

**list\_all = []**

**print('正在获取动漫详细信息...')**

**# 在for循环体中用tqdm()包裹迭代器实现进度条效果**

**for i in tqdm(ids):**

**#将每个id代表的电影详情加入到list\_all中**

**list\_all.extend(get\_movie\_info(i))**

**#将数据转换成json写入到文件中**

**write2json(list\_all)**