哈尔滨工业大学

**<<信息检索>>**

**实验报告**

**(2019年度春季学期)**

|  |  |
| --- | --- |
| **姓名：** | **王丁子睿** |
| **学号：** | **1183710211** |
| **学院：** | **计算机学院** |
| **教师：** | **张宇** |

实验一 网页文本的预处理

## 实验目的

对信息检索中网页文本预处理的流程和涉及的技术有一个全面的了解，包括：

* 抓取网页
* 网页正文提取
* 分词处理
* 停用词处理

## 实验内容

1. 网页的抓取和正文提取

通过爬虫工具爬取网页（至少1000个，其中包含附件的网页不少于100个，多线程实现爬虫可加分），然后提取网页标题和网页正文，以及网页中的附件并保存附件到本地json文件中。

1. 分词处理、去停用词处理

将提取的网页文本进行分词和去停用词处理，并将结果保存。

## 实验过程及结果

1. 核心框架实现

核心框架为utils/base.py中Base类，用于保存爬虫获得的数据，分词，以及将数据保存到指定路径下。

各方法功能如下：

@staticmethod

    def load\_url(url: str,

                 file\_path: str,

                 count: int,

                 ends: List[str] = ('htm')):

        """

        以url为根路径进行BFS，提取count个网页的信息

        经过观察，实验选定的网页的子网页均以.htm作为结尾

        :param url: 检索根路径

        :param file\_path: 网页附件保存地址

        :param count: 检索网页数量

        :param ends: 检索网页特征

        """

def tokenize(self, tokenizer):

        """利用分词器对文本进行分词

        Args:

            tokenizer: 分词器

        """

def save\_json(self, output\_path: str):

        """

        将网页库导出为json文件的形式

        :param output\_path: 输出路径

        """

    @staticmethod

    def load\_json(input\_path: str):

        """从json文件中导入网页库

        Args:

            input\_path: 导入路径

        """

1. 爬虫实现

爬虫程序入口为craw.py，核心代码为utils/page.py。

其中爬虫使用BeautifulSoup，每个Page实例保存一个网页的信息。

爬虫根目录选取哈工大教务处官网（http://jwc.hit.edu.cn/），一个网页的html代码示例如下所示：

文本, 日程表

描述已自动生成

可以发现，可用的子页面网址和可下载链接的属性名均为a href，为了区分和选取需要的下载文件类型，这里只对以.doc、.docx、.txt为后缀的链接进行下载。

此外，每个页面的title直接选取该页面的title属性，paragraphs直接选取该页面的content属性。

为了保证效率，爬取时采用多进程，具体实现方案为：每次搜索到一个可用连接时，便在进程池中放入一个新的爬虫进程，用于获取新连接的数据，进程池会自动调度分配进程。

Page类的各方法功能如下：

    def \_\_init\_\_(self, url, title, paragraphs, file\_name, file\_path):

        """

        声明一个新的网页

        :param url: 网页url

        :param title: 网页标题，字符串或分词列表

        :param paragraphs: 网页正文（取description），字符串或分词列表

        :param file\_name: 附件名称

        :param file\_path: 存储附件的地址

        """

    @staticmethod

    def load\_url(url: str,

                 file\_path: str,

                 code: int,

                 attachment\_type=('txt', 'doc', 'docx')):

        """

        从url中提取信息，并保存到file\_path中

        :param url: 提取url

        :param file\_path: 存储附件的地址

        :param code: 网页编号

        :param attachment\_type: 下载附件类型

        :param tokenizer: 分词器，None不进行分词

        """

    def show(self):

        """

        将该网页以dict的格式返回

        """

    def tokenize(self, tokenizer):

        """

        对该网页的title和paragraphs进行分词处理

        分词结果覆盖原数据

        :param tokenizer: 分词器

        """

1. 分词实现

分词程序入口为segment.py，核心代码为utils/tokenizer.py。

分词器采用LTP，停用词表采用实验所给表，因此没有需要特别说明的地方。

Tokenizer类的各方法功能如下：

    def \_\_call\_\_(self, paragraphs: str) -> List[str]:

        """对给定的文段进行分词处理

        Args:

            paragraphs: 待分词文段

        Returns:

            分词文段

        """

1. 实验结果

实验结果文件位于data/result文件夹中，附件位于data/attachment文件夹中。

实验共统计1037个页面，其中含111个拥有附件的页面。

一个页面信息的例子如下：

{

"url": "http://jwc.hit.edu.cn//2018/0113/c4350a198247/page.htm",

"title": ["哈尔滨", "工业", "大学", "本科生", "交流", "学习", "指南", "（", "2018", "版", "）"],

"paragraphs": ["哈尔滨", "工业", "大学", "本科生", "交流", "学习", "指南", "（", "2018", "版", "）", "哈尔滨", "工业", "大学", "本科生", "交流", "学习", "指南", "（", "2018", "版", "）"],

"file\_name": [],

"file\_path": "Lab1/data/attachment/哈尔滨工业大学本科生交流学习指南（2018版）"

}

其中url为原网页路径，title为原网页标题，paragraphs为原网页正文，file\_name为附件文件名（由于仅收集.doc、.docx、.txt文件，故可能为空），file\_path为附件文件路径。

## 四、实验心得

* 学习了使用BeautifulSoup爬取网页的方法，并使用urlopen下载文件。
* 学习了使用LTP对文本进行分词的方法。
* 学习了python对json文件进行操作的方法。