EE208 Final Project Report

Integrated Search Engine for Elctronic Commodities

电子商品集成搜索引擎

董世文　方少恒　杨弘博　周李韬

GROUP 14

SJTU F1803016

2020 年 1 月 12 日

目录

前言

Part I

Crawler

Chapter 1

京东爬虫

1.1 商品信息提取

dsw

Chapter 2

苏宁爬虫

2.1 网页 URL 爬取

yhb

2.2 商品信息爬取

fsh

2.3 商品评论标签爬取

苏宁的域名组织较有规律，这也是我们选它作为目标网站的原因之一。对于商品详情页，其组织形式是“https://product.suning.com/{}/{}.html.format(shop, sku)”，其中 shop 是 10 位的一串数字，是商铺代号。如苏宁自营是“0000000000”。后面的 sku 是 11 位的商品代号，如 Apple Watch 是“11357287387”。

yhb

2.4 商品评分计算

yhb

Part II

Index & Search

Chapter 3

构建索引

dsw

Chapter 4

图片匹配

4.1 LOGO 匹配

fsh

4.2 LSH 匹配

fsh

Chapter 5

商品检索与排序

前文所述过程中建立的 Lucene 索引表和图像 LSH 哈希表是本章节商品检索与排序的根据，在本项目中，我们实现了两种检索方式——按关键词检索与按图片检索。

按关键词检索部分，用户可以输入商品关键词执行检索，也可以输入商品的品牌等属性执行高级检索。对检索的结果，用户可以选择按照相关度、价格顺序、评分高低进行排序。在图片检索中，用户上传图片后，可以选择 LOGO 匹配或精确匹配两种方式。LOGO 匹配将图片与我们建立的产品品牌图库中的 LOGO 对比，计算 SIFT 特征向量的相似度，返回最有可能的品牌关键词执行关键词检索。精确匹配则将图片特征向量映射到前文建立的 LSH 哈希表中，返回匹配图片的 URL 列表，再到 Lucene 索引表中调用 TermQuery 匹配出 URL 对应的条目，按匹配度返回商品信息到前端中。关键词检索与相关度排序沿用了本课程此前实验的代码，不作详述。本章节将主要介绍高级检索、商品排序和图片匹配三个功能的实现。

5.1 商品高级检索

5.1.1 接口设计

在此前建立索引的过程中，对于所有商品都具备的属性，如品牌、来源网站等，我们建立了专门的可被索引的 Field 进行存储，在本节我们将实现对这些 Field 的多字段检索。

在编写检索程序的代码前，考虑到我们的项目需要整合多种检索和排序方式，包括关键词、多字段检索、价格排序、评分排序等，我们有必要做好统一的接口，实现多重功能在后端查询时的一致。

一方面，用户可能在高级查询页面给出多字段的查询请求，对此我们采取了此前实验中类似的方法，编写了一个 commandParser 将查询条件转换成 BooleanQuery 处理多字段查询的请求。尽管在查询首页中，用户输入的数据是以表单形式传入的，但我们在编写查询程序时仍旧先将其转换成了一条“contents site:... brand:...”的字符串形式。我们之所以不直接提取表单内容，是我们出于下文实现 filter 功能的考量，在 filter 功能中，用户在勾选过滤选项后会再次发出一个包含

搜索的请求，网页需要保存记录此前搜索的字段传入 filter 请求的参数中，为了便于代码的复用，转换成字符串的方式会使程序更加可读和清晰。

另一方面,用户在结果页面可能会选择不同的检索方式,这会决定 Lucene 检索过程中 searcher 的参数，我们需要为此编写不同的搜索函数。在综合考虑整合以上两种因素后，以下是我们为检索程序提供的接口，可以看到，商品检索程序位于 search_command 中，需要提供检索字符串 kw 和 method（默认为按相关度 relativity 排序）两个参数。对获得的商品信息我们会做一些处理，获得 filtertags 等信息，具体将在前端部分进行介绍。

```
# 统一处理搜索请求的web脚本 Web/code.py
class search:
    def GET(self):
        user_data = web.input(website="",brand="")
        kw = user_data.keyword
        if user_data.brand:
            kw += ' brand:%s' %(user_data.brand)
        if user_data.website:
            kw += ' website:%s' %(user_data.website)   # 将输入表单转换成字符串形式的请求
        method = web.input(method="relativity").method.decode('utf-8')
        vm_env.attachCurrentThread()
        contents = search_command(kw,method)      # 搜索结果
        filtertags = total(contents)            # 统计品牌、属性、特色的结果，即显示在页面左侧所必
                                                    须的内容
        results = itemlis(contents)             # 要显示在页面右侧的所必需的内容
        return render.result(kw,method, results, filtertags)
```

5.1.2 多字段查询处理

在检索程序中，我们首先要对输入的 query 字符串进行处理，为此我们编写了一个 command_to_query 函数, 统一处理各种形式的 query。

```
def command_to_query(command,analyzer):
    command_dict = parseCommand(command)          # 将字符串query转换成dict形式
    print command_dict
    seg_list = jieba.cut(command_dict['title'])
    command_dict['title'] = (" ".join(seg_list))  # 对command_dict遍历，建立BooleanQuery
    querys = BooleanQuery()
    for k,v in command_dict.iteritems():
        query = QueryParser(Version.LUCENE_CURRENT, k,
                            analyzer).parse(v)
        querys.add(query, BooleanClause.Occur.MUST)  # 各字段之间为AND关系
    return querys
```

对不同的检索方式，我们设计了如下结构以调用不同的搜索函数。

```
def search_command(query,method):
    ... # 配置searcher、analyzer
    return globals()[method+'_search'](searcher,analyzer,query)
    # 根据method的不同调用不同函数名的搜索函数
```

至此，我们完成了搜索程序接口的设计，并且实现了商品的多字段高级搜索。

5.2 商品按属性排序

对具体的函数设计而言，我们有按相关度排序、按价格排序和按评分排序。相关度排序检测搜索结果与商品名称信息的匹配度，可以调用 Lucene 内置的 scoreDocs 方法实现，在此前的实验中已经完成，不作详述。对价格和评分排序，我们在建立索引时已经将这些信息用 Lucene 内置的 LONG Field 进行存储，我们只需为其建立对应的 SortField，Lucene 就可以实现按排序执行查找，以按评分查找为例，对应搜索程序的脚本如下所示。

```
def rank_search(searcher, analyzer, command):
    rank_sorter = Sort(SortField("score",SortField.Type.LONG,True))  # True表明降序排序
    query = command_to_query(command,analyzer)
    scoreDocs = searcher.search(query, 150, rank_sorter).scoreDocs
    return read_results(scoreDocs,searcher)
    # read_results函数提取搜索结果中的必要信息，将在前端部分介绍
```

5.3 图片匹配

5.4 按图片检索

Part III

Web Front-end

Chapter 6

Web 框架

前端的搭建基于 web.py 框架，在页面美化方面采用了 Bootstrap 框架。下面对 Web 前端的工作作具体介绍。

6.1 web.py 配置

网站前端的 URL 结构如图**??**所示。

6.2 网页功能

6.2.1 搜索主页

6.2.2 商品信息陈列

6.2.3 商品过滤功能

6.3 网页美化

Appendix

写作分配

Acknowledgements

总结