

北京邮电大学



艺术类中文期刊文章信息检索系统

——《信息与知识获取》作业二

班级	学号	姓名	分工
2018211305	2018211290	崔思颖	系统模块设计、数据收集与处理、信息检索模块的设计与实现、UI 设计与实现、作业 2 报告的撰写
2018211305	2018211250	岑明洲	信息抽取模块的实现、作业 3 报告的撰写
2018211305	2018211291	吴川宇	倒排索引文件的生成与查找、作业 1 报告的撰写

2021 年 6 月 7 日

目录

- 实验概述..... 3
- 实验目的..... 3
- 实验环境..... 3
- 系统模块划分 3
 - ____用户输入..... 4
 - ____信息检索..... 4
 - ____相关度计算..... 5
 - ____信息抽取..... 5
 - ____输出 5
- 系统实现..... 6
 - ____数据集 6
 - ____构建倒排索引..... 7
 - ____构建向量空间..... 7
 - ____评分机制..... 8
- 遇到的问题..... 9
- 结果展示..... 12
 - ____数据截图..... 12
 - ____程序运行截图..... 15
- 工程文件说明 20
- 总结..... 21

实验概述

根据老师在课上介绍的关于信息检索和信息提取的算法，参考软件工程所描述的软件系统开发方法，利用 Python 语言，开发简单的搜索引擎系统，主要实现信息检索、信息提取和可视化展示等功能。

实验目的

自己动手设计实现一个信息检索系统，中、英文皆可，数据源可以自选，数据通过开源的网络爬虫获取，规模不低于 100 篇文档，进行本地存储。中文可以分词（可用开源代码），也可以不分词，直接使用字作为基本单元。英文可以直接通过空格分隔。构建基本的倒排索引文件。实现基本的向量空间检索模型的匹配算法。用户查询输入可以是自然语言字符串，查询结果输出按相关度从大到小排序，列出相关度、题目、主要匹配内容、URL、日期等信息。最好能对检索结果的准确率进行人工评价。界面不做强制要求，可以是命令行，也可以是可操作的界面。

实验环境

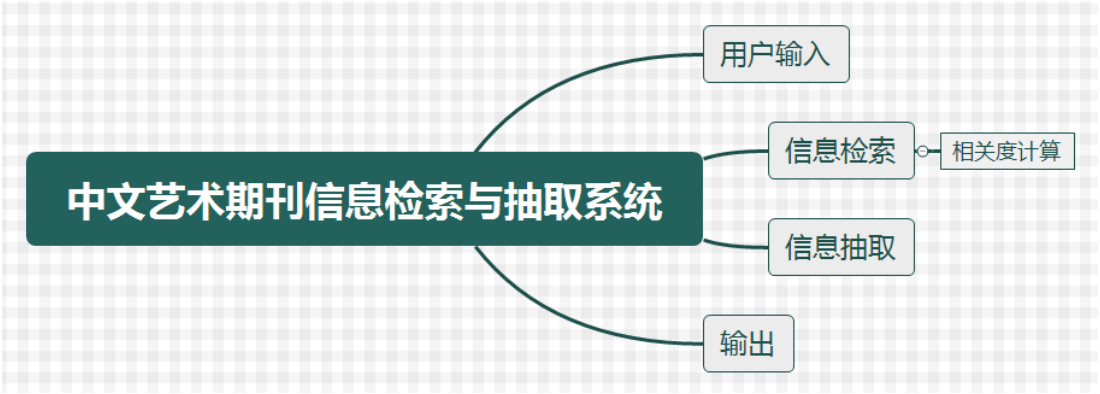
操作系统：Windows 10

集成开发环境（IDE）：PyCharm、QtDesigner

程序设计语言：Python

系统模块划分

本系统主要划分为一下五个功能模块：用户输入模、信息检索模块、相关度计算模块、信息抽取模块、输出模块，其中，相关度计算模块包含在信息检索模块内。模块图如下所示：



用户输入

模块编号	N-01-01
模块名称	用户输入处理模块
模块输入	用户输入的自然语言字串
模块输出	0.txt（用户输入文档文件）
模块功能	将用户输入作为文档以进行后续计算
模块实现	对用户输入进行去空格、分词、去停用词、筛选中文后写入 0.txt 中，作为目标文档。

信息检索

模块编号	N-02-01
模块名称	索引建立模块
模块输入	所有文档
模块输出	倒排索引文件
模块功能	使用文档生成用于查询的倒排索引文件，包含其在文中的词频、位置信息
模块实现	对于每个关键词，找出其在每篇文档中出现的频率、出现的位置，依次写入 index.txt 文件。

模块编号	N-02-02
模块名称	矩阵生成模块
模块输入	包含查询词的集合
模块输出	矩阵 M 、 Pos 。 $M[i][j]$ 表示包含单词 j 在文档 i 中的 $TF*IDF$ 值， M 的最后一行表示用户查询内容的向量； $Pos[i][j]$ 为单词 j 在文档 i 中的位置列表。
模块功能	将相关结果以矩阵的形式加入程序中进行计算
模块实现	调用 <code>sklearn</code> 库，将分词后文档中的词语转化为词频矩阵，并将词频矩阵统计成 $TF-IDF$ 值。

相关度计算

模块编号	N-03-01
模块名称	相关度计算模块
模块输入	矩阵 M
模块输出	以相关度从大到小排序的（文档序号，相关度）列表 Sim
模块功能	计算每篇文档与查询内容的相关度并排序输出
模块实现	调用 <code>sklearn</code> 库中的 <code>cosine_similarity</code> 函数，计算目标文档与其他文档的余弦相似度，按相关度值从大到小排列

信息抽取

模块编号	N-04-01
模块名称	实体识别模块
模块输入	文档编号
模块输出	实体识别结果
模块功能	将检索结果中的文档进行实体识别，进行特定信息点抽取后，输出
模块实现	调用 <code>pkuseg</code> 工具包的 <code>cut</code> 方法进行词性划分和实体识别，以及改写 <code>cocoNLP</code> 工具包中的 <code>extractor</code> 模块的源码来实现信息抽取

输出

模块编号	N-05-01
模块名称	信息检索输出模块
模块输入	列表 Sim, Pos 矩阵, 全部文档
模块输出	用图形界面打印按相关度从大到小排序的查询结果, 包括相关度、题目、日期、版号、作者、主要匹配内容等信息
模块功能	将信息检索的结果进行输出
模块实现	在 UI 界面中新建文本框与按钮, 并对布局进行设计, 每个文本框中首先输出包含关键词的文档头部, 再通过倒排索引文件查找到关键词的位置, 输出主要匹配内容。

系统实现

数据集

我们选择了现有的复旦大学中文语料库作为数据源，它由复旦大学李荣陆提供。我们选取了其中 500 篇艺术领域的文本作为检索对象，文本示例如下：

【文献号】3-385
【原文出处】文艺理论研究
【原刊地名】沪
【原刊期号】200101
【原刊页号】13~20
【分类号】J2
【分类名】中国古代、近代文学研究
【复印期号】200107
【标题】中国古代的文艺本体论
【作者】蔡鍾翔
【作者简介】中国人民大学中文系
【正文】

在进入本论题之前,有必要对“本体”这个概念作一点辨析。记得数年前在文艺报刊上曾讨论过“本体”滥用的问题。如《文艺报》1995年42期刊登的黄力之的《“本体”的滥用及其文化意义》,指出了这种情况。滥用的始作俑者可能是王蒙同志,他在《读评论文章偶记》(载《文学评论》1985年6期)一文中有这样一段话:我以为,我们更应该重视对文学的本体论的研究。对文学的本体的提法的科学性我并没有把握,我请求读者和专家原谅我知识的不足和用语的大胆。但我以为文学的本体是存在的,它就是文学所反映所追求所赖以发生的宇宙、自然、世界、人生、社会、生活、人类的精神世界,它也就是古往今来古今中外的文学作品、文学宝库本身。王蒙同志的看法是有意义的,但他解释的文学本体,确实不符合哲学上本体范畴的涵义,其实他所指的文学中所反映的内容、文学的本质特征以及文学作品本身等等,是不能用“本体”这个概念来概括的。确如上述那篇文章说的,这样的所谓“本体”就成了“大而无当的东西”了。后来《文学评论》1996年第6期又发表了朱立元教授的论文《当代文学、美学研究中对“本体论”的误释》。他举出了五种误释:一、把“本体”误作“本身”。二、把“本体”误释为世界万物的“本源”或“本性”。三、把“本体论”与“宇宙论”混淆起来。四、把“本体性”与过程性、体验性、自足性、根本性等意义相混淆。五、把哲学本体论与西方存在主义哲学的联系割断,他还指摘了中国哲学史学者汤用彤、张岱、熊十力的误释。朱教授的意见应该说是中肯的,但他是严格的以西律中,则未免有些绝对化。我认为,西方文化与中国文化是异质文化,西方的概念、范畴引进或移植到中国来是难免会走样的,往往要同中国原有的概念、范畴相比附,使之本土化,这种走样或“误释”有些是可以容许的,当然我们并不主张滥用。张岱年先生已经认识到西方的“本体”范畴在中国哲学中是没有的,他在《中国哲学大纲》一书中指出:

印度哲学及西洋哲学讲本体,更有真义,以为现象是假是幻,本体是真。本体者何?即是唯一的究竟实在。这种观念,在中国本来的哲学中,实在没有。中国哲人讲本根与事物的区别,不在于实幻之不同,而在于本末、原流、根茎之不同。万有众象同属实在,不惟本根为实而已。以本体为唯一实在的理论,中国哲人实不主持之。(注:《中国哲学大纲》(北京:中国社会科学出版社,1982年),页9。)因此张先生就采用了“本根”这一传统概念,以示与西方的区别。但汤用彤先生等也没有错,因为ontology译为“本体”也是借用了传统概念,中国哲学中就有“本末体用”之辨,那末也可以有中国的“本体”范畴和“本体论”。《中国大百科全书·中国哲学卷》“本体论”一条将中西本体论分别疏解,不失为一种通达的处理。本文所论即是就中国哲学的本体论而言,但不把它解为“大而无当的东西”。

“本体”是什么?是指宇宙万物的本原或最高的抽象存在。“本体论”则是指探究宇宙万物存在的终极根据的理论。中国古代的哲学本体论,主要有三种观点,即以道为本的道本论,以气为本的气本论,和以心为本的心本论(以理为本的理本论,可以归属于道本论)。这几种本体论,决定了文艺学中的不同的文艺本体论;文艺本体论可以说是从哲学本体论推行出来的,而文艺本体论则对文艺理论产生深刻的影响。不了解古代的文艺本体论,就很难理解和把握古代文艺理论的基本精神。以下分别论述三种文艺本体论及其在文艺理论中的影响。

构建倒排索引

所谓倒排索引，就是按照索引去反向查找文件的过程。其思路为从单词角度看文档，标识在文档中每个单词分别出现了多少次（词频）及其出现的位置（相对于该文档首部的偏移量）。

创建倒排索引，分为一下两个步骤：

1. 创建文档列表

2. 创建倒排索引列表：对文档中的数据进行分词，得到词条。然后记录下包含该词条的所有信息为 `index.txt`，文件的格式如下：

```
1. word1 fre1（在文档 1 中的频次）,pos1,pos2... fre2（在文档 2 中的频次）,pos1,pos2...
2. word2 fre1,pos1,pos2... fre2,pos1,pos2...
3. word1 fre1,pos1,pos2... fre2,pos1,pos2...
```

构建向量空间

我使用 Python 的工具包“`sklearn`”来构建文档的空间向量。

“`sklearn`”使用词袋和 TF-IDF 模型来表示文本数据，其中，词袋模型对应的就是文档向量。为了方便实现，我们使用词袋模型来构建空间向量，也就是说，我们只关注文档中是否出现给定单词以及单词出现的频率，而舍弃文本的结构、单词出现的顺序和位置。

一般来说，对于一个文本语料库，构建词袋有一下四个步骤：

1. 文本分词：把每个文档中的文本进行分词

2. 构建词汇表：把文本分词得到的单词构建为一个词汇表，包含文本语料库中的所有单词，并对单词进行编号。假设词汇表

有 n 个单词，单词编号从 0 开始，到 $n-1$ 结束，可以把单词编号看作是单词的索引，通过单词编号可以唯一定位到该单词。

3. 词向量表示：每个单词都表示一个 n 列的向量，在单词编号（词汇索引）位置上的列值为 1，其他列的值为 0

4. 统计频次：统计每个文档中每个单词出现的频次。

我们使用“sklearn”即可快速构建模型，核心代码如下：

```
1. def GetTFIDF(corpus):
2.     '''生成 TFIDF 矩阵'''
3.     vectorizer=CountVectorizer()          #将文本中的词语转换为词频矩阵
4.     X=vectorizer.fit_transform(corpus)    #计算每个词语出现的次数
5.     words = vectorizer.get_feature_names()
6.     transformer=TfidfTransformer()
7.     M=transformer.fit_transform(X)        #将词频矩阵 X 统计成 TF-IDF 值
8.     return words,M.toarray()
```

评分机制

我们将查询语句作为一个伪文档加入文档集，构造一个长度为 $N+1$ 的文档列表，List 后 N 项是 N 篇文档的内容字符串，第一条是查询语句。sklearn.feature_extraction.text 包下的 CountVectorizer 类可以通过 List 计算出文档集的词频矩阵，词频矩阵的每一行表示文档集中的一个文档，词频矩阵的每一列表示文档集经过分词操作得到的一个词，词频的计算方法如下所示

$$tf_{ij} = \text{count}(\text{word}) / \text{count}(\text{document})$$

$\text{count}(\text{word})$ 表示该词在文档中的出现次数、 $\text{count}(\text{document})$ 表示该文档的长度， i, j 分别词频矩阵的第 i 行和第 j 列。

`sklearn.feature_extraction.text` 包下的 `TfidfTransformer` 类可以通过文档集的词频矩阵计算出该文档集的 TF-IDF 矩阵，矩阵的每一行表示文档集中的一个文档，矩阵的每一列表示文档集经过分词操作得到的一个词，其中 TF 为词频、IDF 为逆文档频率。IDF 值的计算公式如下所示 (N 为文档集的文档数，df 为该词在文档中出现的文档数)：

$$idf_{ij} = \log \frac{N}{df}$$

TF-IDF 值计算过程如下：

$$(tf-idf)_{ij} = tf_{ij} \times idf_{ij}$$

TF-IDF 可以评估一个字词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在语料库中出现的频率成反比下降。所以可以用 TF-IDF 矩阵中的某一行向量来作为表示这一行所代表的文档的向量，通过依次计算 TF-IDF 矩阵第一行的行向量和后 N 行的行向量之间的余弦相似度即可得到衡量查询语句和文档集中的一篇文档的相似度的权重（两个向量的余弦越大，它们的夹角越小，它们之间的距离越近）。此外，若发现文档标题中包含关键词，则每包含一个关键词，相关度加 1。最后将结果返回给 GUI 数据展示模块（相似度大于 0 的才允许加入返回结果 List）。代码实现如下：

```
1. def CalSim(M, words):
2.     '''计算相关度'''
3.     Sim=[]
4.     for i in range(1, len(M)):
```

```

5.         sim=cosine_similarity([M[0]], [M[i]])           #计算余弦相似度
6.
7.         '''找到文档标题'''
8.         f=open('docs_utf8\\'+str(i)+'.txt', 'r', encoding='utf-8')
9.         head=f.readline()
10.        while(not ('【 标 题 】'in head)):
11.            head=f.readline()
12.        f.close()
13.
14.        '''文件标题中每有一个关键词，则相关度加一'''
15.        for word in words:
16.            if word in head:
17.                sim[0][0]+=1
18.
19.        Sim.append((i,sim))
20.        Sim.sort(key=lambda x:x[1][0][0],reverse=True)    #按相关度从大到小排
    序
21.        return Sim

```

遇到的问题

● 文本文件编码杂乱

实验最初对语料库中的 txt 文件进行读取时，我们发现四百多篇文档编码不一，有 GBK、ANSI、UTF-8 等等，导致无法正常读取。我们使用了 python 的 chardet 工具包对每篇文档的编码进行识别，然后进行相应的解码读取，最后再统一用 UTF-8 编码写回文档中。其中有多篇文档的编码识别不出来，我们对其编码进行了手工的修改。

● 倒排索引文件中词语位置不正确

生成 index.txt 倒排索引文件后，进行相关内容一项的输出，发现输出的结果不对，于是文件进行了检查，最后发现是词语后的多了第一列，因为在最开始测试时我将用户输入文件作为 0.txt 写入

原文档目录下，导致后来生成 index.txt 把 0.txt 也当成数据源了，多输出了一系列不必要的索引。

● 索引结果总是为 0

在程序运行成功后，输入查询，总是查找不到相关文档，调试发现是因为输出的时候没有用用户输入的分词结果，而是用了原输入，导致无法在索引文件中找到对应词。将用户输入的分词结果保存作为输出模块的输入后，问题解决。

● python 中无法提前声明全局变量

之前用 C++ 写 GUI 时，会将界面指针与其他全局变量声明在前，但 python 中没有指针，信息抽取窗口只有在知晓文档编号时才能创建，而函数执行完该窗口又会被回收。解决办法是，定义一个全局字典变量，在新建窗口时，为字典新添键值对，就可以将窗口的内存一直保留到程序运行结束。

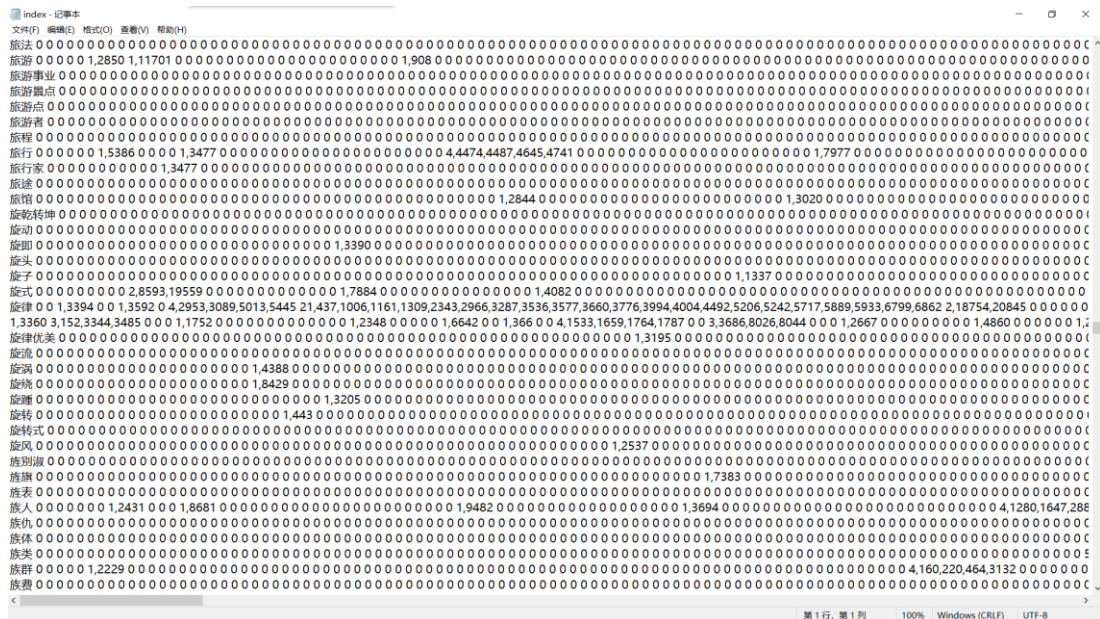
● PyQt 库中的 ButtonGroup 信号与槽函数问题

由于 "Extract" 按钮与文本框的数量是动态变化的，所以无法为每个按钮提前绑定好槽函数，上网查阅资料发现，可以使用 ButtonGroup 类解决此问题，在为每个按钮设置 ID 后加入到 ButtonGroup 中，在按钮被点击时就可以获取其 Id，运行初，Id 值错误，返回了 -1，对代码反复调整并调试后，找到原因是单个按钮也在 UI 中绑定了点击触发的槽函数，导致 ButtonGroup 的点击信号被覆盖，没有返回的 ID 值，删去后信号触发槽函数返回的 ID 正确。


结果展示

数据截图

● 倒排索引文件



● 停用词文件

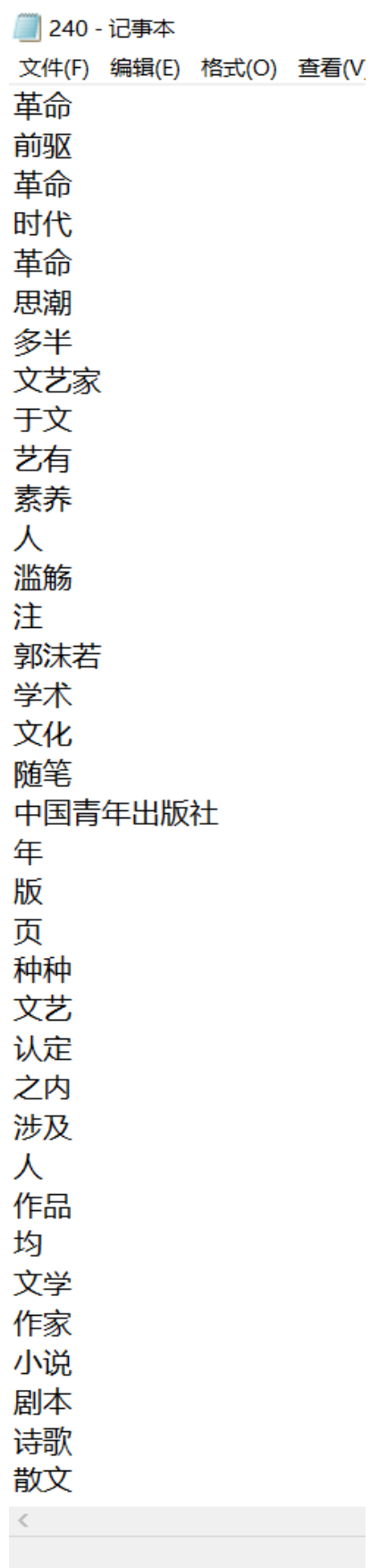
 stop_words - 记事本

文件(F) 编辑(E) 格式(O) 查看(V)

已经
至于
後面
之后
纵然
例如
哈哈
今后
有利
今年
其它
广大
逐渐
果然
正常
正在
本着
各自
若非
使用
如何
现在
既然
另外
可能
这点
总结
时候
不够
完成
甚至
下列
先後
不单
上面
看出

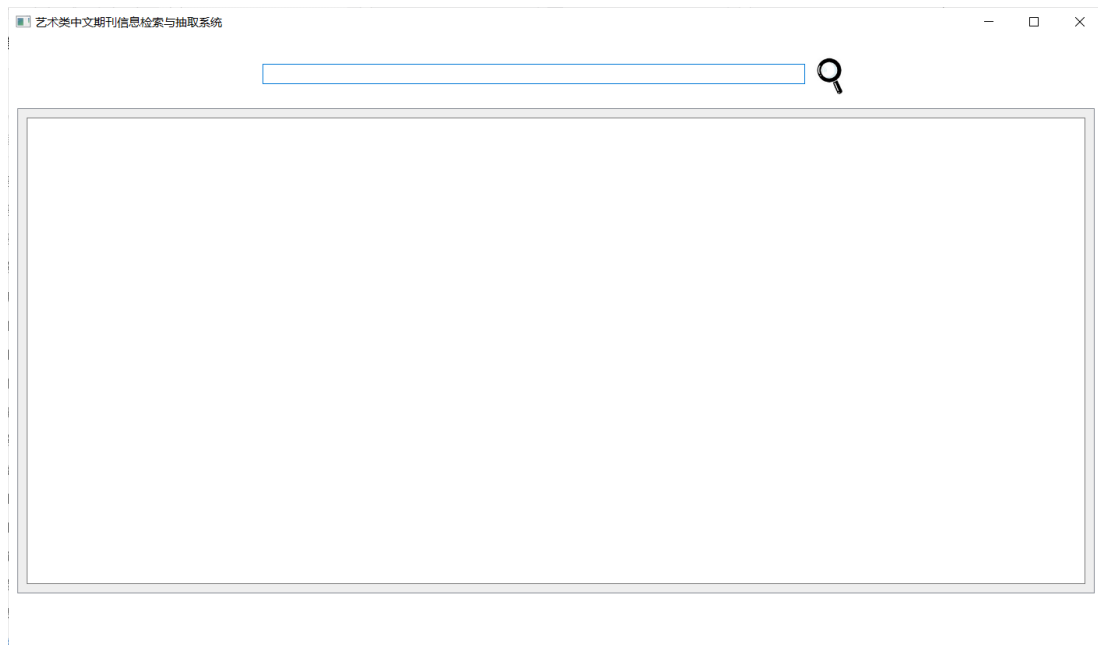
<

● 分词、去停用词处理后的文档内容

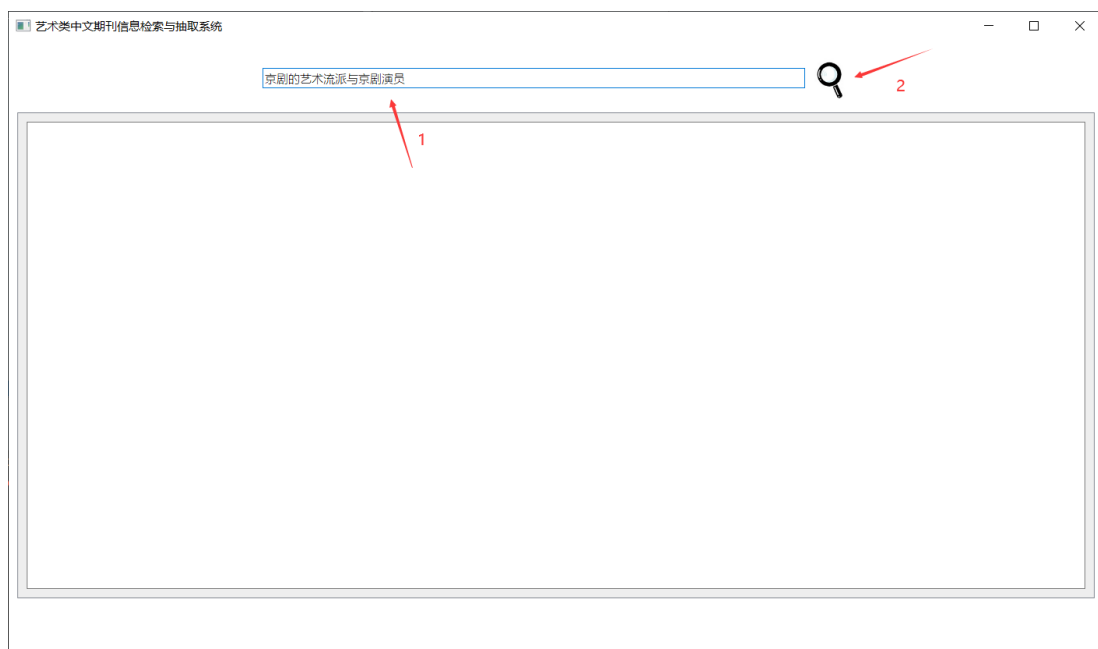


程序运行截图

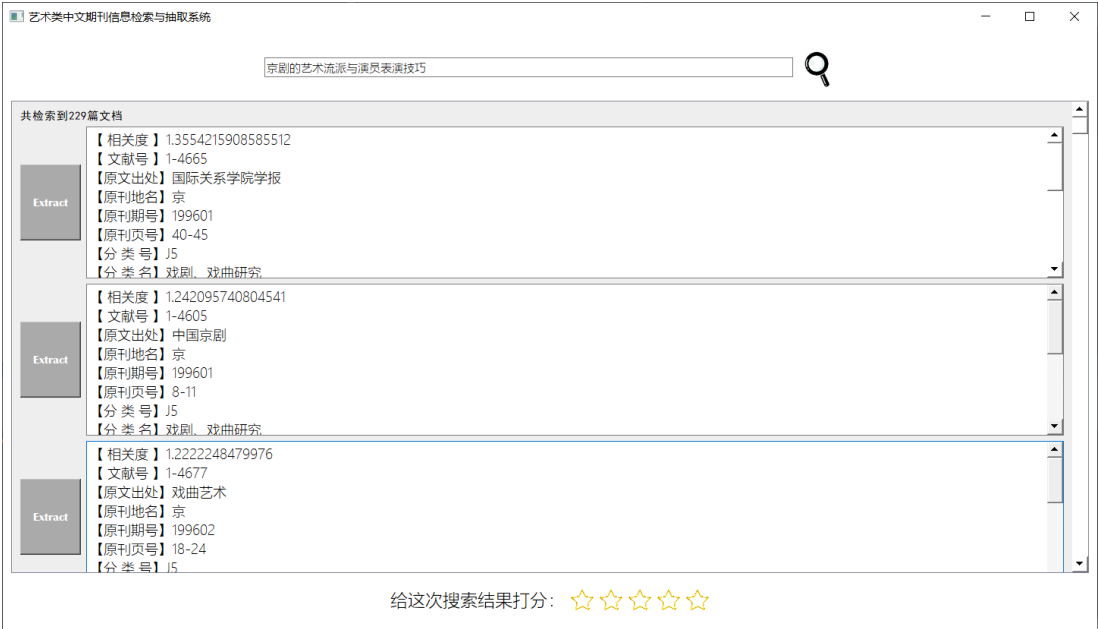
1. 安装相关库，运行 search_GUI.py 文件，主界面如下



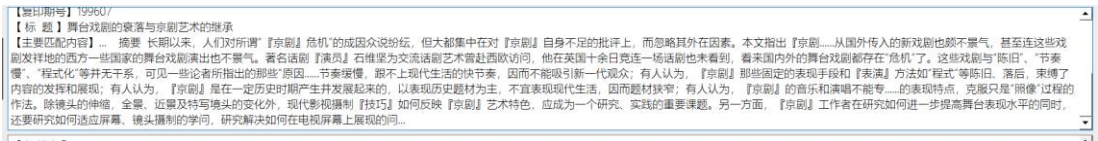
2. 在上方条形文本框中输入想要查询的内容，点击放大镜标志进行搜索



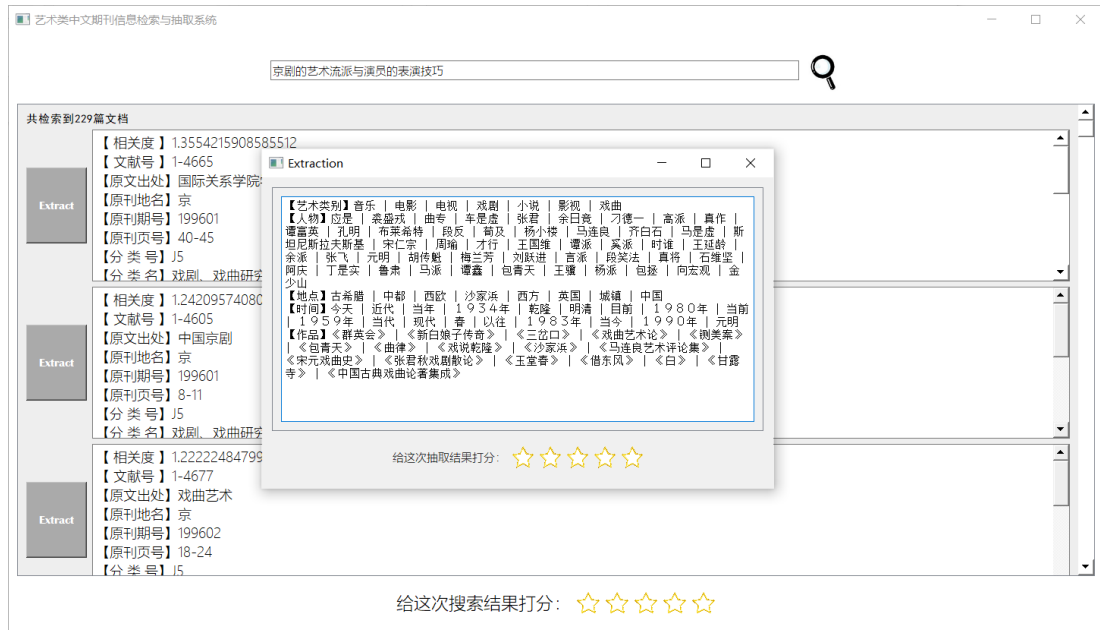
3. 搜索结果显示于中间的滚动区域，按照相关度从大到小排列，并且显示了检索到的相关文档总数为 229 篇。



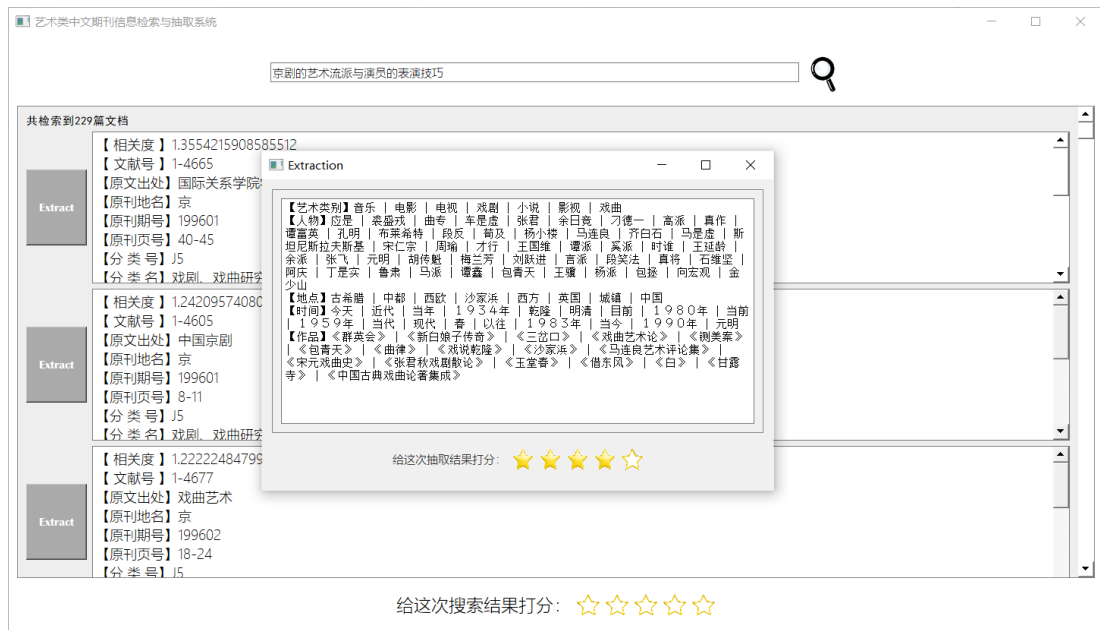
4. 其中，每篇个文本框里显示了相关文档的相关度、文献号、标题等信息，下拉可以看到主要匹配内容，搜索内容中的关键词用“『』”强调



5. 点击搜索结果文档项旁侧的 Extract 按钮，即可实现对该文档的正文进行信息抽取，抽取的兴趣点包括艺术类别、人物、时间、地点、书名，抽取的实现具体细节请看作业 3 报告。



6. 点击 Extraction 窗口下方打分的五角星按钮，可以对搜索结果进行人工评价。点击过后按钮失效，避免重复评分。



7. 评价的结果会写入项目文件夹中的“extraction_rating.csv”

文件中，第一列为文档编号，第二列为评分，便于以后对于不同的文档抽取结果的准确度进行统计评价，从而改进系统。

	A	B
	71	5
	111	5
	147	5
	64	4

8. 点击界面下方打分的五角星按钮，可以对搜索结果进行人工评价。

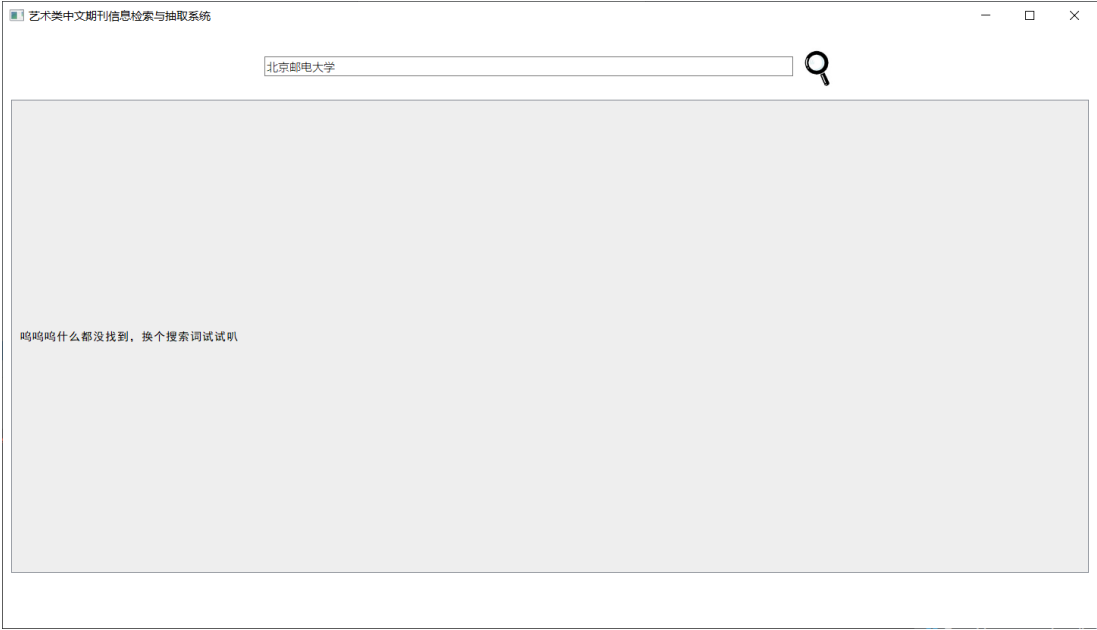
点击过后按钮失效，避免重复评分。



9. 评价的结果会写入项目文件夹中的“search_rating.csv”文件

中，第一列为搜索内容，第二列为评分，便于以后对于不同的搜索内容的准确度进行统计评价，从而改进系统。

11. 如果查询无结果，则会显示未找到



工程文件说明

文件/文件夹	说明
docs_utf8	原始文档文件夹，所有文档已改为 UTF-8 编码并从 1 到 500 标号。
clean_data	包含所有文档进行分词、去停用词、去英文后的纯净文本
cocoNLP、stanfordcorenlp、stanford-corenlp-4.2.2	进行信息抽取需要用到的开源代码库，我们对其中的源码进行了修改
img	图形化界面使用的所有图片文件夹
extract_widget.ui	信息抽取小窗口的用户界面
GUI.ui	程序主界面
index.txt	倒排索引文件
source.qrc	图形化界面的资源文件
stop_words.txt	停用词表

word_split.py	对文档和用户输入进行分词、去停用词等处理
build_index.py	读文档，构造空间向量，生成倒排索引文件，相关度计算
extract.py	信息抽取
output.py	输出索引结果中的文档头和主要匹配内容
search_GUI.py	含 UI 界面的窗口类以及所有槽函数，窗口 UI 的构建，程序的主要逻辑
source_rc.py	.qrc 资源文件经工具处理后转化的.py 文件

总结

本系统针对艺术类中文期刊文章构建信息检索系统，同时构建信息抽取模块抽取多个兴趣点，信息抽取模块的具体细节将在作业三的报告给出。

其中系统主要实现功能：文档数据的处理、倒排索引文件的生成、空间向量的构建、文档相关度的计算、文档信息的抽取、操作界面的图形化。

系统的数据源采用了复旦大学语料库中 500 篇艺术类别中文期刊文章。信息检索算法的实现通过计算文档的 tf-idf 值来进行计算，将输入问题作为文档看计算文档的 tf-idf 向量，再通过计算其和每个具体景点信息的余弦相似度来排序。图形界面的设计采用 PyQt 库实现，PyQt 是一个创建 GUI 应用程序的工具包。它是 Python 编程语言和 Qt 库的成功融合。Qt 库是目前最强大的库之一。

最终本系统实现上述功能，能够实现文档的检索和抽取，并且有友好的操作界面，可以通过搜索栏去检索到与检索语句有关的文档

信息，同时能够对结果进行人工评价。还可以通过点击 Extract 按钮进行信息抽取，并对抽取结果进行评价。

在进行实现系统的过程中，遇到许多问题和挑战，例如怎么将信息检索算法从理论变成现实，在数据的获取中怎么能够保证高效的获取数据，数据获得后怎么去进行输出与展示等。极大地锻炼了我的编程能力，并对《信息知识与获取》这门课有了更深的理解与感悟。