摘要

自从 google 公司推出旗下的产品 Knowledge Graph 以来，知识图谱这个概念越来越受到学术与工业界的关注。如何以质量层次不齐的网页数据作为原始数据源，构建知识图谱已经成为了一个热门的研究课题。

本文首先介绍本课题的研究背景，主要任务与目标：构建基于文本的知识图谱；然后介绍本课题相关的技术背景：知识图谱以及文本情感分析；接着介绍实验的数据处理和构建知识图谱的算法设计：本文使用 Beautiful Soup 进行文本提取，使用 jieba 工具进行分词，使用流水线方法进行实体关系抽取；随后本文介绍具体的实验过程，给出实验结果和实验评估；最后本文对本课题的工作做一个总结和展望。

关键词：知识图谱 情感分析

作　　者：徐双奇

指导老师：朱晓旭

Abstract

Since google company introduced its product Knowledge Graph, the concept of knowledge graph has been increasingly concerned by academic and industrial circles. How to use the Web page data with uneven quality as the original data source to build a knowledge graph has become a hot research topic.

This paper first introduces the research background, main tasks and goals of the project: constructs a text-based knowledge map; then introduces the related technical background of this topic: knowledge graph and text sentiment analysis; then introduces the experimental data processing and algorithmic construction of the knowledge graph design: This paper uses Beautiful Soup for text extraction, and uses jieba tools for word segmentation, and the use of pipeline method for entity relationship extraction; then this paper describes the specific experimental process, gives experimental results and experimental evaluation; Finally, this paper summarizes and prospects the work of this topic.

Keywords: knowledge map sentiment analysis

Author: Shuangqi Xu

Instructor: Xiaoxu Zhu

第一章　概述

1.1 引言

互联网的发展导致了“数据大爆炸”。但正所谓"乱花渐欲迷人眼”，大量的数据反而导致人们很难从中获得有效信息。于是诸如谷歌、百度之类的搜索平台在上世纪九十年代应运而生,他们一般采用向量空间模型(VSM)。然而这些基于关键词匹配的搜索技术目前存在一些缺陷:搜索引擎缺乏互联网网页中人，物，事，之间的联系，因此无法完成精确搜索。

于是知识图谱这一概念应运而生。它将人物，事物作为实体，并构建他们之间的关系。数据不再是孤立的个体，相反充满了联系。

目前，面向特定领域的中文知识图谱的项目比较缺乏。因此，构建一个特定领域的知识图谱具有重要现实意义。

我国的互联网正在蓬勃发展中。截至 2018 年初，我们国家的互联网用户已经颇具规模，达到近八亿人。网民人群数量的急剧增长改变了我国传统监督体系。人可以通过互联网在各大论坛上面发表自己的意见，提出自己的不满。基于互联网的舆论监督正在发挥着越来越重要的作用。

本文的工作关注互联网用户投诉这一特定领域，以投诉文本作为原始数据，构建基于投诉文本的知识图谱。

1.2 课题的主要任务和目标

本课题的主要内容是（1）对投诉文本进行分词，从而为之后提取实体以及他们之间的关系提供基础。（2）从投诉文本中提取实体概念。（3）从投诉文本中提取实体概念之间的关系。（4）用实体概念以及他们之间的关系构建知识图谱。本课题的主要任务是基于用户投诉文本进行知识图谱的构建,即输入一个关键词输出与该关键词相关联的信息。并以视觉(图片)的方式呈现这些信息。

1.3 本文的组织

本文的第一章是概述，主要阐述课题背景以及课题的主要任务与目标。本文第二章介绍相关技术背景:一是知识图谱技术背景，二是文本情感分析技术背景。第三章

介绍数据整理和算法的设计:首先是数据清洗的方法，接着是中文分词，然后是停用词处理，最后是知识图谱的提取。第三章介绍具体实验设计，分析，和评估方法。第五章对本课题做一个总结并且对课题做进一步展望。最后是参考文献和致谢。

第二章　相关技术背景

2.1 知识图谱

2012 年谷歌发布了一款重磅产品“知识图谱”（Knowledge Graph）。知识图谱可以将谷歌的结果进行知识系统化，用户所输入的任何一个关键词都能获取完整的知识体系。比如你输入特朗普，谷歌的侧栏就会告诉你特朗普的相关信息，比如他的身份，配偶，以及其子女信息等等。换句话说在知识图谱的概念里面，一切都是一个实体，而非简单的，没有联系性的字符串[1]。事实上知识图谱是大数据时代的必然产物，互联网上爆炸的数据使得人们迫切需要一个将这些数据关联起来的工具。

知识图谱本质上是一个语义网[2]。语义网络最早可以追溯的上个世纪六十年代，它是一个有向图，这个有向图由顶点和边构成。顶点代表实体或概念，有向图中的边代表实体或概念之间的关系。在早期，语义网络的构建一般由手工构建。知识图谱的数据源是互联网上的网页数据，包括结构化，半结构化，以及非结构化数据。其中大部分数据都是非结构化的。这些数据需要通过数据清洗等手段进行数据的预处理，因为这些原始数据里面充满了噪声比如歧义等等。

大体上，知识图谱的构建可以分为以下几个部分：一是实体或概念的获取；二是关系的提取。三是实体或概念与关系的关联，去除伪信息以及互斥信息，构建知识图谱。常见的实体或概念与关系的识别方法有两种:一是流水线方法[3]，二是端到端识别方法。关于实体关系的去除伪信息以及去除互斥信息。最近提出了一种可行的方法是使用概率软逻辑[4]，它既可以捕获事实之间的概率依赖关系又在实体和其关系之间执行全局约束。根据应用场景，实体或概念与关系的存储可以采用关系型数据库以及 NoSQL。如果使用场景需要高性能计算，则需要配合 Hadoop 或者 Spark 使用。

2.2 文本情感分析

文本情感分析又被称为意见挖掘[5]，是指对存在主观情感倾向的文本进行情感分析和提取的过程。文本情感分析技术一般被用来做网络营销、企业舆情监控、政府舆论监控等等用途。一般来讲，文本情感分析技术可以划分成几个子任务，首先是提取给定文本的主题。接着是确定陈述者主体，然后是提取该主体的陈述的主观性语句，过滤那些客观性语句。最后是对他的情感倾向进行判断分析。根据给定文本的粒度不

同，可以将文本情感分析划分为词汇级，语句级，篇章级，以及海量数据级。针对海量数据级，随着互联网的发展，网络的各个平台都存在“网络水军”，他们所陈述的意见往往都是不真实的，可能会对最终结果造成较大影响。因此需要首先根据噪声的特征进行数据清洗。

常见的文本情感分析方法有：（1）基于机器学习的技术，比如支持向量机等等。（2）基于词典的方法，即对一个新出现的词，根据词典中相似词对其进行情感倾向推理。目前，中文的文本情感分析还存在一些挑战，具体在:（1），效果良好的英文文本情感分析技术无法在中文文本上获得相同效果。(2)，不同语境下的情感分析技术效果不佳，等等。随着智能手机的火热以及诸如微博这类信息发布和分享平台的发展，短文本的情感分析将会越来越受到人们的重视。

第三章　数据整理和算法设计

3.1 数据的清洗

本课题的数据是“苏州阳光便民 12345”网站的投诉网页，格式是 HTM 如图 3.1 所示。

图 3.1. 苏州阳光便民 12345 网页。

“苏州阳光便民 12345”是由苏州市人民政府组织创建的网站。它成立于 2005 年，网站的建站目的是通过整合政府部门和有关行业单位的公共服务资源，设立统一的服务热线平台，365 天×24 小时为市民提供方便、快捷、优质、规范的服务。

市便民服务中心为市政务办下属、副处级、公益一类事业单位。主要职责：一是负责拟订全市便民服务工作的标准、规范，并组织实施；二是负责市“12345”政府公共服务热线的运行管理，承担与其他市级公共服务热线和服务平台的对接、联动、合作；三是负责市“12345”政府公共服务热线的诉求受理、答复，事项的交办、跟踪、督办和回复回访，协助做好承办单位的业务协调、服务质量和服务效果的监督考核工作；四是依托“12345”服务平台，开展社情民意调研、舆情分析，向公众提供相关的政务信息服务；五是参与文明创建、社会信用、城市管理、城市综合服务标准化、政风行风、作风效能、绩效管理等公共事项的管理；六是参与防汛防旱、防台风、突发水污染、突发固体废物污染、食品安全等突发事件的应急管理；七是指导各市、区便民服务业务工作。

可以从图 3.1 看到该 HTM 网页中的本文所需要的投诉文本：（1）投诉标题；（2）投诉正文（3）网友针对投诉内容的跟帖与回复。

为了提取这三类文本，需要观察该 HTM 的源码。三类文本都在 HTM 网页固定的标签之中：对于投诉标题，如图 3.2 所示，它总是位于 H1 标签中最里面的位置。而投诉正文和跟帖回复则位于属性的开头为“postman”的 TD 标签之中，如图 3.3 以及图 3.4。

<H1 class=ts><A href="forum.php?mod=forumdisplay&fid=2&filter=typeid&typeid=3">[咨询投诉]</A> [苏州市] <A id=thread_subject href="forum.php?mod=viewthread&tid=41157">常熟张桥染整厂烟囱排出来的刺鼻性气味是高致癌物质吗？</A> </H1><SPAN class=xgl><A onclick="return copyThreadUrl(this)" title=您的朋友访问此链接后，您将获得相应的积分奖励 href="forum.php?mod=viewthread&tid=41157">[复制链接]</A> </SPAN></TD></TR></TBODY></TABLE>

图 3.2　投诉标题位于 H1 标签最里面

<TD id=postmessage_464535 class=t_f>常熟市幸庄镇张桥区集镇上有4、5家染整厂，每天烟囱里排出大量刺鼻性气味，我也打电话到环保局投诉过，要求提供具体地址具体厂家。请问那位大虾能告知下，染整厂排出的刺鼻性气味是否是高致癌物质，如何取得证据及投诉维权？？先谢谢大家。
</TD></TR></TBODY></TABLE></DIV>

图 3.3　投诉正文位于属性的开头为“postman”的 TD 标签之中

<TD id=postmessage_464666 class=t_f>您反映的问题已关注。</TD></TR></TBODY></TABLE></DIV>

图 3.4　投诉回复也位于属性的开头为“postman”的 TD 标签之中

从 HTM 格式的数据中提取特定标签内容的工具有很多，本文使用比较常见的工具：Beautiful Soup。利用该工具提取出投诉文本之后，本文需要对其做进一步处理。

3.2 中文分词

为了从文本中提取实体或概念与关系，本文首先得对投诉文本进行中文分词。所谓分词就是将连续的字符串或序列按照一定的规范重新组合成词序列的过程。在英文的行文中，单词之间是以空格作为自然分界符的，因此，在词理解上就比较直观。而汉语中词与词之间没用分界符，因此需要本文进行分词。

分词的算法有很多，总体可以分为两类：一类是无词典的分词方法，另一类是有词典的分词方法。所谓无词典就是根据汉字串在汉字语料中出现的概率来推断出该汉字串是否为词。有词典的分词方法是根据现有词典进行分词的方法。常见的有词典分词算法有：（1）正向最大匹配算法；（2）逆向最大匹配算法；（3）双向匹配法；（4）邻近匹配算法；（5）最短路径匹配算法；（6）基于统计的最短路径分词算法。

正向最大匹配算法的思想是以中文字符串的首字为起点，找到在中文字符串中出

现的最长的词。这即是该字符串的第一个分词，剩余的字符串按照相同的方法处理。

逆向最大匹配算法的做法和上面的正向对最大匹配算法相反，它以汉字字符串的尾部为起点，找到最长的词汇，然后切分。并对剩余的汉字字符串做相同的动作。

双向匹配算法是将正向最大匹配算法和逆向最大匹配算法结合起来，构成新的算法。

邻近匹配算法是一种改进的最大匹配算法，它改进了了正向最大匹配算法的效率。

最短路径匹配算法是将汉字字符串进行全切分，接着构建汉字字符串切分的有向无环图。每一个汉字分词对应一条边，每条边上有一个权值。找出从有向图的起点到终点的最短路径，该路径包含的词就是汉字字符串的切分结果。

基于统计的最短路径分词算法是对最短路径匹配算法的一种改进，它根据词频给词典里的词分配不同的权重，具体的权重值可以通过大规模语料库获得。

常用的分词工具有很多，本文选择比较常见的 jieba 分词工具。Jieba 分词工具所采用的分词算法便是上述的基于统计的最短路径分词算法。在有向无环图中求最短路径，把分词问题变成用动态规划解决的最短路径问题。对于未登录词，jieba 采用了基于汉字成词能力的 HMM 模型，使用维特比算法。

3.3 停用词处理

经过上一步的分词处理，投诉文本变成了汉字词汇序列。但是这些词汇序列中还存在着停用词，本文需要对其进行过滤。停用词就是停止处理的词语，因为这些词语没有实际的含义。却占用存储空间，并且会对结果产生影响。停用词的过滤一般有以下几种方法：（1）将数据集中的高频与低频词汇找出，人工提取有用的词汇，剩余的就是停用词。（2）参考互联网上的常用中文文本停用词表。

本文将使上述两种方法结合起来，先参考哈工大停用词表，再对高低频词汇人工筛选。

3.4 知识图谱提取设计

知识图谱中的结点代表实体，边代表实体之间的关系。接下来本文需要做的是实体识别和关系抽取。

在本章的前面几小节里面，本文对投诉文本进行分词，停用词过滤。一组中文词

汇集合。为了进行实体识别，本文对该词汇集合中的所有词按照其词频从大到小进行排列，选取词频较高的词语，把他们当作实体。关于关系抽取，常见的主要有两种方法。一种是流水线方法[6]，另一种方法是基于神经网络的联合学习[7]。

流水线方法的思想是：第一步，将实体与实体之间两两组合。第二步，进行关系分类。

与流水线方法不同，联合学习对实体识别和关系抽取使用单一模型，比如使用 LSTM 模型等等。

本文使用实现简单的流水线方法，首先进行实体识别，再进行关系抽取。

3.5 本章小结

本章具体讨论了构建基于投诉文本的中文知识图谱的步骤以及各个步骤所使用的算法：首先从 HTM 格式网页提取投诉文本。本文使用 Beautiful Soup 来提取这些文本。接着对投诉文本进行中文分词。这一步本文使用的工具是 jieba 分词。随后本文对词汇集进行停用词过滤。针对不同的主题，停用词也不尽相同。本文采用了参照停用词表与人工筛选相结合的方法，对停用词过滤。最后本文进行提取知识图谱。本文采用的是流水线方法，将问题分成两个子任务：实体识别和关系抽取。对于实体识别，本文采用两两组合配对的方法。

第四章　实验设计分析和评估

4.1 实验设计

如图 4.1 所示，本文的实验设计可以划分为以下几个步骤：（1）从网页中提取投诉文本；（2）对投诉文本进行分词；（3）停用词过滤；（4）提取高频词作为实体；（5）计算实体相关性；（6）构建知识图谱。

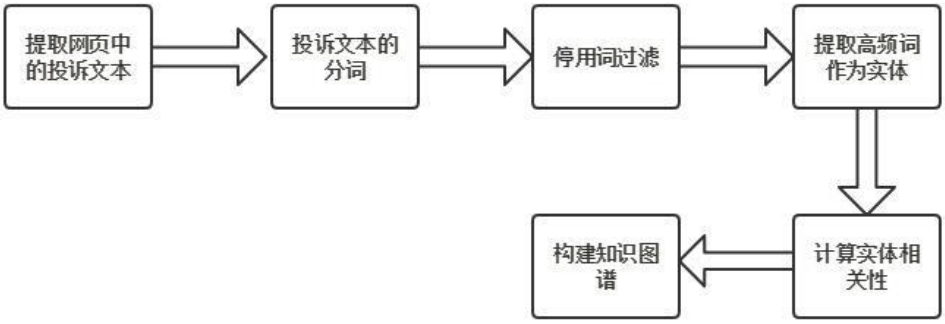

图 4.1 实验步骤

首先本文使用 Beautiful Soup 提取 HTM 文件中投诉文本，投诉文本又可以分成投诉标题，投诉正文，以及投诉回复三个部分。本文分别编写不同的代码提取之。

对于投诉标题，由于它位于 H1 标签中最里面的位置。因此本文采用以下主要代码提取它：

```
from bs4 import BeautifulSoup #导入包
import re
import os
list=os.listdir("../html")
f=open("../text/投诉文本.txt","w",encoding='utf-8')#utf-8 方式打开
#print(len(list))
for i in range(0,len(list)):
    path = os.path.join("../html", list[i])
    t="../text/htm/"+str(i)+".txt"
    if os.path.isfile(path):
        soup = BeautifulSoup(open(path, 'rb'))  # 以 rb 的方式打开文件
        tf=open(t,"w",encoding='utf-8')
        headlist = soup.find_all('h1')
        for head in headlist:
            alist = head.find_all('a')
```

```
            for a in alist:
                if ("[咨询投诉]" not in a.string):
                    f.write(a.string)
                    f.write("\n")
                    tf.write(a.string)
                    print(a.string)

        # 接着提取正文和回复 <TD id=postmessage。。。。>
        tdlist = soup.find_all('td', attrs={'id': re.compile("postmessage")})
        for td in tdlist:
            text = td.get_text()
            f.write(text)
            tf.write(text)
            f.write("\n")
            print(text)
        tf.close()
        print(i)
f.close()
```

在上面代码的第 14 行，本文过滤了“咨询”，“投诉”这两个停用词。原因是短语“咨询投诉”在每个 HTM 网页中固定位置都会出现，因此失去了实际的意义。

对于投诉正文和投诉回复,它们都位于属性的开头为“postman”的 TD 标签之中，因此可以用下面的代码提取：

```
tdlist = soup.find_all('td', attrs={'id': re.compile("postmessage")})
for td in tdlist:
    text = td.get_text()
    f.write(text)
    tf.write(text)
    f.write("\n")
    print(text)
```

本文通过上述两份主要的代码从 HTM 网页中提取投诉文本，并将投诉文本写入 txt 格式文件中。

接下来，本文从这个 txt 格式文件中读取投诉文本，使用 jieba 分词工具对其进行分词。并用停用词表进行停用词过滤。最后按照词频从高到低对词语排序，选取前 1500 个词语。主要代码如下：

```
import jieba
import re
from collections import Counter
f=open("../text/投诉文本.txt",'r', encoding='UTF-8')#只读，注意编码方式
```

```
of=open("../text/高频词.txt",'w', encoding='UTF-8')
ef=open("../text/实体.txt",'w',encoding='UTF-8')
jieba.load_userdict('../text/自定义词.txt')#导入自定义词
#加载停用词表
stopWord = [line.strip()for line in open('../text/停用词.txt').readlines() ]
line=f.read()
if line:
    seg_list = jieba.cut(line, cut_all=False)
    c = Counter()
    for x in seg_list:
        if len(x) > 1 and x != '\r\n'and (x not in stopWord )and not x.isdigit():
            if(not re.match(r'[a-zA-Z]+',x)):#正则表达式过滤英文
                c[x] += 1
    for i in c.most_common(1500):
        of.write(str(i[0])+' '+str(i[1])+'\n')
        ef.write(str(i[0]) + '\n')
        print(i)
f.close()
of.close()
ef.close()
```

经过观察，jieba 对于极个别未登录词的分词效果不好，例如词语“寒山闻钟”被划分成了“寒山闻”。因此需要在 jieba 中导入自定义词：

```
    jieba.load_userdict('../text/自定义词.txt')#导入自定义词
```

同时，高频词中存在诸如“&#”，英文字符串，数字，等等无意义的词语，这些都没有被过滤掉。解决的方法是在停用词表里添加停用词，如图 4.2，并且配合正则表达式进行过滤：

```
    for x in seg_list:
    if len(x) > 1 and x != '\r\n'and (x not in stopWord )and not x.isdigit():
        if(not re.match(r'[a-zA-Z]+',x)):#正则表达式过滤英文
            c[x] += 1
for i in c.most_common(1500):
    of.write(str(i[0])+' '+str(i[1])+'\n')
    ef.write(str(i[0]) + '\n')
    print(i)
```

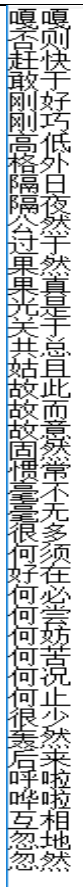

图 4.2　停用词部分截图

为了对处理结果有一个感性认识，本文使用了 matplotlib 库来进行数据可视化，以柱状图的形式展示高频词的词频分布情况，下面是主要代码：

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['simHei']
plt.rcParams['axes.unicode_minus']=False#解决中文乱码问题
file=open("../text/高频词.txt",'r',encoding='UTF-8')
eef=open("../text/实体关联.txt",'r',encoding='UTF-8')
ees=open("../text/实体关联排序.txt",'r',encoding='UTF-8')

eesdic={}
eesline=ees.readline()
while eesline:
    text=eesline.strip('\n')
    eesdic[(text.split()[0],text.split()[1])]=text.split()[2]
    eesline=ees.readline()
freq=[]
k=0
for i in eesdic.values():
    freq.append(float(i))
```

```
    k+=1
    if k>=15000:
        break

plt.bar(range(len(freq)),freq,color='gray')
plt.show()
file.close()
eef.close()
ees.close()
```

接下来，本文将进行实体识别和关系抽取。为了探究实体之间的相关性，本文将之前的前 1500 个高频中文分词两两组合，并且计算这两个实体在同一个 HTM 格式的投诉网页上出现的频率。经过实验，单独按照某两个实体在网页中一同出现的频率降序排列并不能准确描述实体与实体之间的关联性。本文采用了如下公式：

$$K = \frac{P(A,B)}{P(A)*P(B)}$$ 公式 4.1

上面公式 4.1 中的 *P(A)*表示实体 A 在数据集中出现的概率，P(B)表示实体 B 在数据集中出现的概率。而 P(A,B)则表示实体 A 和实体 B 同时出现在某个投诉网页上的概率。本文将计算 K 值并降序排列得出关联性最强的实体，主要代码如下：

```
eef=open("../text/实体关联.txt",'r',encoding='UTF-8')
ef=open("../text/高频词.txt",'r',encoding='UTF-8')
kf=open("../text/实体关联排序.txt",'w',encoding='utf-8')#utf-8 方式打开
eedic={}
edic={}
eesdic={}
line=eef.readline()
while(line):
    text=line.strip('\n')
    eedic[(text.split()[0],text.split()[1])]=text.split()[2]
    #print(eedic)
    line=eef.readline()

line=ef.readline()
while line:
    text = line.strip('\n')
    edic[text.split()[0]] = text.split()[1]
    line=ef.readline()
for i in eedic:
    eesdic[(i[0],i[1])]=int(eedic[i])*33526/int(edic[i[0]])/int(edic[i[1]])

sort=sorted(eesdic.items(),key = lambda x:x[1],reverse = True)
for i in sort:
```

```
    kf.write(str(i[0][0])+' '+str(i[0][1])+' '+str(i[1])+'\n')
    print(i)
eef.close()
ef.close()
kf.close()
```

接下来，本文将编写应用程序来展示实验的结果，该应用程序的功能是：用户通过在输入框内输入中文字符串，并点击搜索按钮，应用程序便将该字符串当作实体，返回与该实体相关的信息。本文使用了 tkinter 库来编写图形界面程序，该应用程序的主要代码如下：

```
import tkinter as tk
from tkinter import *
file=open("../text/实体关联排序.txt",'r',encoding='UTF-8')
dic={}#存放数据的字典
line=file.readline()
while(line):
    text=line.strip('\n')
    dic[(text.split()[0],text.split()[1])]=text.split()[2]
    line=file.readline()
print(len(dic))
delete=['客户']
def searchEntity(text):
    for i in dic:
        if text==str(i[0]) and str(i[1]) not in delete:
            print(str(i[1]))
            strvar.set(str(i[1]))
            return
        elif text==str(i[1]) and str(i[0]) not in delete:
            print(str(i[0]))
            strvar.set(str(i[0]))
            return;
    print("NONE")
    strvar.set("NONE")
def button():
    t=search.get()
    searchEntity(t)
def labelClear():
    t=search.get()

gui=tk.Tk()
gui.title('基于投诉文本的知识图谱')
gui.geometry('500x300+700+300')
search=tk.Entry(gui)
go=tk.Button(gui,text='GO',command=button)
```

```
clear=tk.Button(gui,text='CLEAR',command=labelClear)
strvar=StringVar()
strvar.set("")
label=tk.Label(gui,textvariable=strvar)
search.pack()
go.pack()
label.pack(padx=5, pady=80)
gui.mainloop()
file.close()
```

4.2 实验结果分析

本小节将要分别展示（1）提取投诉文本；（2）投诉文本分词以及停用词过滤并且提取高频词；（3）实体相关性计算；（4）搜索功能应用程序，这几个部分的结果。

首先是投诉文本的提取，如图 4.3 所示：

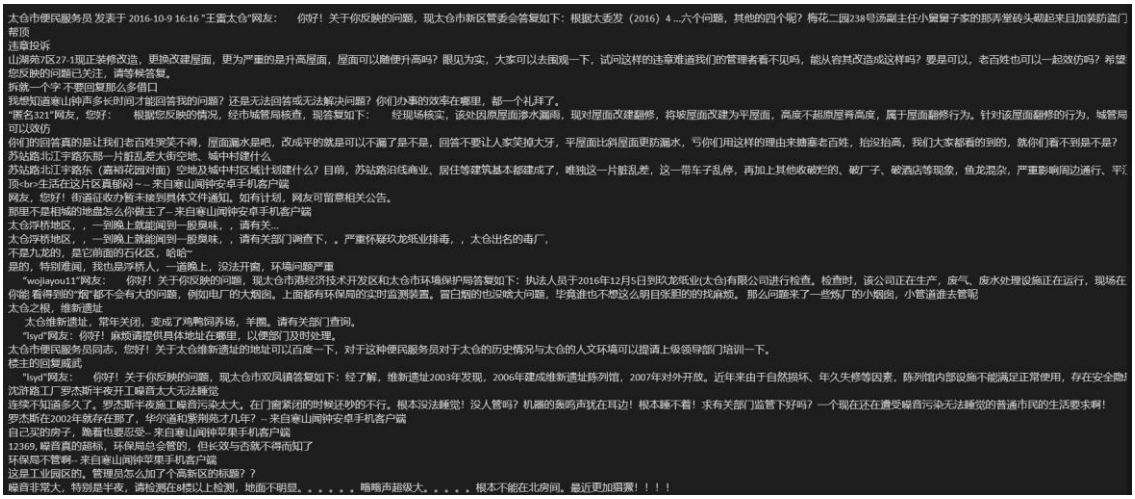

图 4.3　投诉文本提取结果

可以在上图中看出，用 BeautifulSoup 工具从苏州阳光 12345 网站的网页中提取投诉文本效果比较良好，基本完成了对投诉标题，投诉内容，以及投诉回复这几部分内容的提取，而且没有提取其他无关的内容。

投诉文本的分词，停用词的过滤以及高频词的提取的结果如图 4.4 以及 4.5 所示：

施工 8093
严重 7942
加强 7898
一下 7880
花园 7829
楼主 7796
通过 7796
道路 7785
发现 7674
这种 7568
一直 7551
工作人员 7531
相城区 7522
吴江 7461
房屋 7350
存在 7098
针对 7009
园区 7005
建议 6851
安全 6728
企业 6604
告知 6543
高新区 6421
老百姓 6367
无法 6329
学校 6328
及时 6226
中队 6005
环境 5975
申请 5970
正常 5887
房子 5875
调查 5803
区域 5794
具体 5782
使用 5767
派出所 5741
沟通 5740
核实 5732
物业公司 5560
垃圾 5554
行为 5534
医院 5515
违章 5497
吴中 5468
正在 5443
职能部门 5440
交警 5407
造成 5327
中心 5324
周边 5294
拆迁 5280

图 4.4 部分高频词截图

图 4.4 的第一列是分词结果，第二列是词频。上图显示用 jieba 分词工具分词，并且结合停用词表进行停用词过滤。导入自定义词，最后得出的高频词结果比较合理。过滤了无意义的英文词语，以及数字符号。

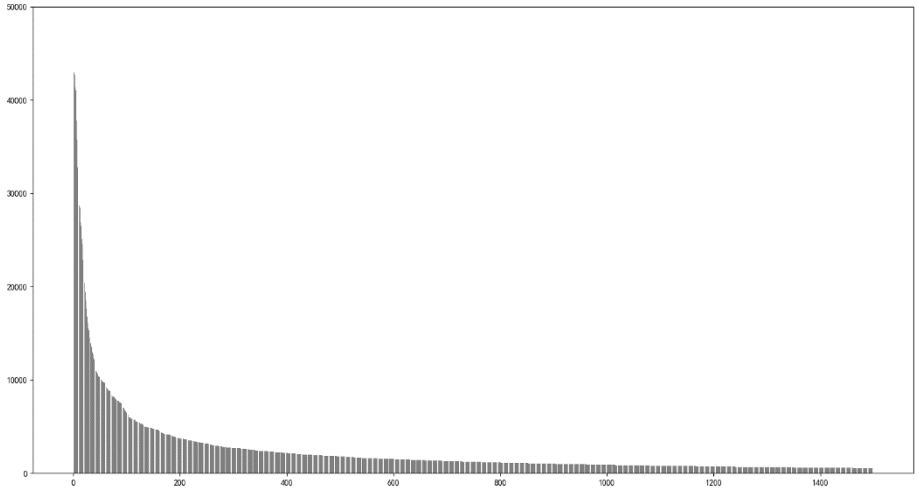

图 4.5　高频词的词频分布

图 4.4 的横轴代表前 1500 个高频词，纵轴代表这些高频词所对应的词频。该图显示了高频词的词频从高到低的变化：前 200 个词语的词频急速下降，而后面的词语的词频缓慢下降最终趋于平稳。

根据公式 4.1，可以得出实体与实体之间的相关程度。本文将实体与实体序列按照它们之间的相关度从高到低排列，结果如图 4.6 以及图 4.7 所示：

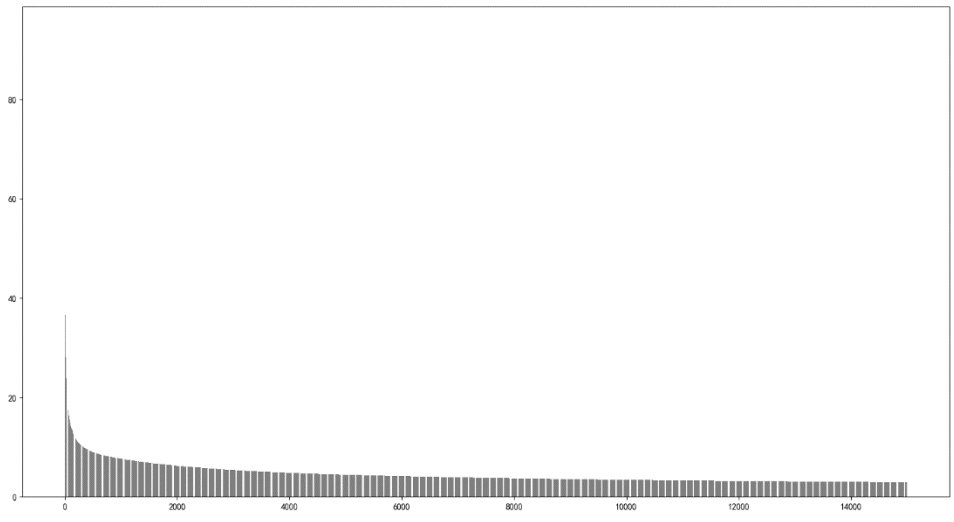

图 4.6　实体相关性公式 K 值分布

客户 法规 15.106632059854514
谅解 敬请 15.020486683757614
信号 绿灯 14.982534542630662
左转 绿灯 14.887694354099583
信号灯 绿灯 14.77219238336891
驾驶员 驾驶 14.753642036124795
非机动车 机动车道 14.60395711126
一步 不可 14.58172949966603
客户 高度 14.550903401418866
公安局 公安 14.525055775032884
产证 国土局 14.403984948313893
东山镇 东山 14.379655674586552
第一 第一次 14.35231815604395
客户 一月 14.296860215757853
直行 绿灯 14.281238167360849
客户 民生 14.259324293108078
客户 市政 14.243732646883712
交管 绿灯 14.16909652490022
拨打 可拨打 14.150194449595647
驾驶 遵守 14.142491943873539
开发 城区 14.098013566527154
客户 有个 14.039149457572924
客户 道理 14.03772857891474
客户 遵守 13.992054012749378
隐患 火灾 13.923734038193894

图 4.7　实体关联性排序部分截图

上图显示的是前 15000 个实体与实体序列的 K 值分布情况。该图显示：前面的一小段的 K 值下降十分迅速。随后 K 值趋于平稳。

最后，本文展示搜索功能的应用程序，其界面如图 4.8：

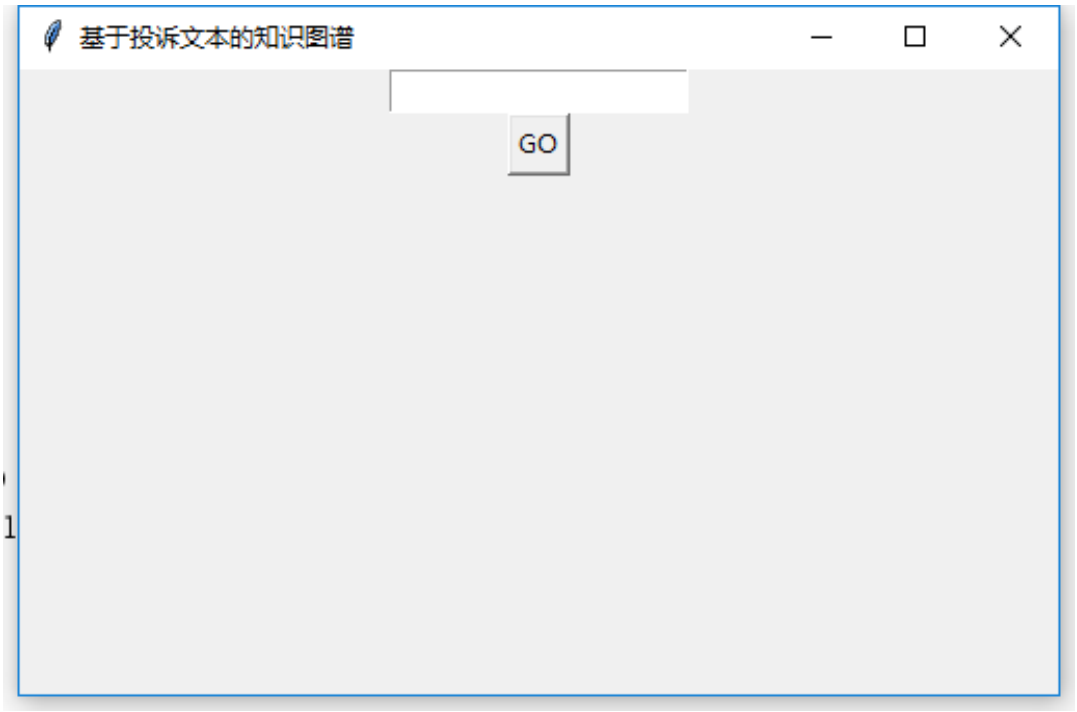

图 4.8 应用程序界面截图

在上图的输入框里输入中午字符串，应用程序得到相应结果。如图 4.9：

图 4.9 搜索结果截图

上图显示：当用户输入“非机动车”中文字符串时，应用程序将该字符串当作实体，返回与之相关的实体“机动车道”。

当用户输入的字符串被当作实体后找不到相关联实体，应用程序结果如图 4.10 所示：

图 4.10　应用程序搜索不到相关联实体截图

4.3 实验评估

本文的实验数据是 33526 个 HTM 格式网页。本文以图片的形式展示基于投诉文本的知识图谱结果。如图 4.11。

图 4.11 知识图谱截图

可以从上图看出，大体上基于文本的知识图谱中的实体关联比较准确。但是仍然存在一些关联度较低的实体组合被错误识别为高关联度等问题。这是本课题以后需要改进的方向。

4.4 本章小结

本章首先阐述了实验的整个流程步骤。接着进行具体的实验设计：（1）用 Beautiful Soup 工具提取投诉文本。该步骤的提取准确率高，为接下来的步骤奠定了良好的基础。（2）用 jieba 分词工具对上一个步骤的文本数据进行分词，并使用停用词表对划分好的词语进行过滤，之后提取高频词语。使用 matplotlib 库，以直方图的形式展示高频的词语的词频分布。该步骤中，在被提取的高频词里仍然存在这一些没有实际含义的词语，比如在图 4.4 中，“一下”就是没有实际含义的词语。本文使用的停用此表没能很好的过滤。（3）使用流水线方法进行实体识别和关系抽取。这种方法的编程语言实现较为简单但是准确率不高，容易出现累计错误问题。（4）最后本文通过使用 tkinter 库编写应用程序，以视觉的方式展示了基于投诉文本的知识图谱的搜索应用。本章的最后进行实验评估，对 33526 个网页数据进行实体关系抽取，并用视觉方式展

现结果。

第五章　总结和展望

5.1 课题总结

联系具有普遍性。大数据时代下，如何把互联网上良莠不齐的数据联系起来成了一个热门的研究课题。而其中知识图谱技术便是一个很好的解决方案。知识图谱将互联网上的“字符串”当作实体，构建他们之间的关系。

本文首先介绍课题的内容，任务，以及目标。接着，本文介绍课题的技术背景：第一是知识图谱的技术背景；第二是文本情感分析的技术背景。随后本文介绍搭建基于投诉文本的知识图谱的数据整理算法以及知识图谱提取算法。接下来，根据这些算法，并且结合诸如 Beautiful Soup，jieba 分词，matplotlib，等等工具进行具体的实验，取得了预期的实验效果。

5.2 课题展望

虽然本课题实验取得了一定的效果，但是由于知识图谱构建技术正在发展之中，技术不是非常成熟。再加上个人对知识图谱构建和相关技术的理解并不十分深刻，以及课题研究时间紧迫等等原因。本文的相关工作目前仍然存在一定问题，下面详细阐述这些问题。

一是停用词过滤效果不是十分理想，可能的原因是本文的数据属于特定主题：投诉类文本，而本文选择停用词表是通用停用词表。另一个可能的原因是本文的数据是网络数据，与书面类型的文本数据相比文法结构缺乏严谨。因为停用词过滤的效果较差，印象了后续部分的实验结果。一种可能有效的方法是使用两种类型的停用词表进行停用词过滤，一种表是投诉类型停用词表，另一种表是网络停词表。将这两个表结合起来可能会对来源于网络的，投诉类型的数据的停用词过滤有显著效果。

二是在实体识别部分，本文采用的是流水线方法。该方法的实现较为简单，但是流水线方法存在很多问题：首先是准确率不高，因为流水线方法是将实体与实体两两组合，这样就出现了冗余，从而导致了较低的准确率。使用流水线方法的另一个弊端是算法执行效率低，也就是流水线方法的时间复杂度较高，为 $O(n^2)$。一种改进的方法是使用联合学习的算法。

三是知识图谱程序的展示结果形式比较单一，不具备美观性。可以改进的措施是

放弃使用应用程序，改用 web 页面展示结果。同时使用互联网上成熟的前端框架，保证页面的美观性。

参考文献

[1] Singhal A. Introducing the knowledge graph: things, not strings[J]. Official google blog, 2012.

[2] sowa J F. Principles of semantic networks: Exploration in the representation of Knowledge[J]. Frame Problem in Artificial Intelligence, 1991(2-3):135–157.

[3] Young D M. Pipelined method and apparatus for processing communication metering data: U.S. Patent 6,377,939[P]. 2002-4-23.

[4] Pujara J, Miao H, Getoor L, et al. Knowledge graph identification[C]//International Semantic Web Conference. Springer, Berlin, Heidelberg, 2013: 542-557.

[5] Pang B, Lee L. Opinion mining and sentiment analysis[J]. Foundations and Trends® in Information Retrieval, 2008, 2(1–2): 1-135.

[6] Young D M. Pipelined method and apparatus for processing communication metering data: U.S. Patent 6,377,939[P]. 2002-4-23.

[7] Zheng, Suncong, et al. "Joint entity and relation extraction based on a hybrid neural network." Neurocomputing 257 (2017): 59-66.

致谢

我要感谢我的指导老师朱晓旭老师，他所讲授的《中文信息处理》课程生动活泼。在课堂上，他深入浅出的授课风格给我留下了深刻的印象，激发了我对自然语言处理的兴趣。他对学生认真负责，每周都会组织一场例会，与我们探讨当前工作进展，并指出下一步的方向。在朱老师身上，我看到了很多老师所缺失的师风师德。他是我学习的榜样，向他致敬！