基于snownlp的词云分析系统实现

摘 要

因特网使得信息的采集、传播的速度和规模达到空前的水平，但与之俱来的问题和“副作用”是：汹涌而来的信息有时使人无所适从，从浩如烟海的信息海洋中迅速而准确地获取自己最需要的信息，变得非常困难。

该系统使用python的django框架;使用snowNLP等算法模型，通过网上浩如烟海的已被标记的被购买商品评论(如淘宝评论，京东评论)数据作为训练数据，进行情态算法训练，自动生成0到1之间的情态评分；使用python的baiduspyder爬取百度的相关数据作为源数据。

通过词云分析系统，能对信息进行提纯，不用在可以直接找出最相关的词语，还通过情态分析将好的评论与坏的评论分开。

**【关键词】**爬虫；中文分词；语义分析；可视化页面

**AWord cloud analysis system based on snoonlp**

**Abstract**

The Internet makes the speed and scale of information collection and dissemination reach an unprecedented level, but the accompanying problems and "side effects" are: the surge of information sometimes makes people at a loss, and it becomes very difficult to quickly and accurately obtain the information they need most from the vast sea of information.

The system uses the Django framework of Python; Using the algorithm model such as snownlp, we train the modal algorithm by using the massive online marked reviews of purchased goods (such as Taobao reviews and Jingdong reviews) as the training data, and automatically generate the modal score between 0 and 1; Use Baidu pyder of Python to crawl the relevant data of Baidu as the source data.

Through the word cloud analysis system, the information can be purified, and the most relevant words can be found directly without using the word cloud analysis system.

[Key words] reptile; Chinese word segmentation; Semantic analysis; Visualization page

**1 绪论**

1.1课题研究的背景及意义

随着科技的发展，人类知识的倍增周期不断缩短，信息量的翻涌几何级别增加，有报告称：“近30年来，人类生产的信息已超过过去5000年信息生产的总和。”除了感叹信息爆炸，每个人的注意力都被不断分割，只能够获取或被获取一定量的信息。如何从海量信息中快速高效地获取优质信息，就成为更为重要的事情。因为低质的无用信息不仅侵占我们的时间，还浪费我们的精力。如今我们身处信息爆炸的大环境，各个公众号、微博、知乎等等，同样地在瓜分我们有限的时间和注意力。罗胖在今天的跨年演讲说到，大意是，各个产品竞争的本质实际上是在争夺用户的时间。因而，当我们每天面对各种各样的信息时，我们要善于进行挑选，而不是一股脑地被动接受。我们急切的渴望通过一种工具能可视化的得到一个东西的所有客观评论，而不是沉没在数据海洋里迷失方向。

1.2国内外情绪分析库介绍

大数据能否真正的具备参考价值？国外的情绪分析库有很多，正确率很高，相比之下国内的情绪分析就比较薄弱了。未来科技之争必定在于对大数据的处理，对数据的处理是给人看的，是可视化的，是符合人心的。世界万千数据几乎全是人为创造的，只要是人创造的数据都暗含情绪。我们处在一个数据爆炸的年代，随着存储设备和互联网的发展，数据已经远远人们所能处理的极限了，利用机器对数据进行处理是必然的。只有这样才能充分发挥大数据的优势，也只有机器才能处理数以兆计的数据并进行统计。如今普通的机器处理分类已经不能满足人们的需求，对数据进行情绪分类，并指导人们进行决策已经势在必行。现在中文情绪分析库有：jiagu、

Snownlp、cemotion、sentiment。

**Jiagu：**

Jiagu使用大规模语料训练而成。将提供中文分词、词性标注、命名实体识别、情感分析、知识图谱关系抽取、关键词抽取、文本摘要、新词发现、情感分析、文本聚类等常用自然语言处理功能。参考了各大工具优缺点制作。

**Snownlp：**

SnowNLP是一个python写的类库，可以方便的处理中文文本内容，是受到了TextBlob的启发而写的，由于现在大部分的自然语言处理库基本都是针对英文的，于是写了一个方便处理中文的类库，并且和TextBlob不同的是，这里没有用NLTK，所有的算法都是自己实现的，并且自带了一些训练好的字典。

**Cemotion：**

Cemotion是Python下的中文NLP库，可以进行 中文情感倾向分析。Cemotion的模型经 循环神经网络 训练得到，会为 中文文本 返回 0～1之间的 情感倾向置信度。您可以批量分析中文文本的情感，并部署至Linux、macOS、Windows等生产环境中，无需关注内部原理。该模块依赖于TensorFlow环境（会自动安装），较老的机器可能无法运行。

**sentiment：**

Sentiment是一个node.js模块，使用‎‎AFINN-165‎‎单词列表和‎‎表情符号情绪排名‎‎对任意输入文本块进行‎‎情绪分析‎‎。情绪提供了几件事情：‎

‎从AFINN单词列表中附加和覆盖单词/值对的能力‎。

‎轻松添加对新语言的支持的能力‎。

‎能够根据每种语言轻松定义否定、强调等的定制策略‎。

1.3课题研究的主要内容

本文主要对百度数据库的相关数据中的数据项进行一一爬取，分析，可视化、在线更新。同时使用4种常用的情绪分析库进行分析并允许动态改变关键词、好词评分下限、坏词评分上限、（这几种算法都将词语的情绪分为0.0~1.0数字越小越消极）生成图云可视化。方便用户对比四种算法的区别。词云分析系统分为以下五个部分：数据采集与数据分析处理、可视化页面显示。下面将针对各个部分进行详细的介绍。

（1）数据采集

对采集到的数据进行分析和设计，所以数据的来源是本系统需要解决的一个问题。数据采集的方法有很多种，比如相关网站提供的API接口，针对于特定网站按照一定的关键词所写的爬虫程序等。爬虫部分主要工作是负责从关键词相关网站上利用网页采集器采集百度评论。我们使用的是baiduspider库，BaiduSpider是一个爬取百度搜索结果的Python爬虫，目前支持百度网页搜索，百度图片搜索，百度知道搜索，百度视频搜索，百度资讯搜索，百度文库搜索，百度经验搜索和百度百科搜索同时还支持多页，多参数搜索。

（2）分析处理

通过jiagu、Snownlp、cemotion、sentiment、模型对数据进行情态评分，分数越高越积极，越低越消极。我们可以通过搜索页面对词云图的积极数据范围和消极数据范围划界。因为网上积极情绪较多，cpu性能等的制约，消极分数不能太低，否则数据太少。

（3）可视化页面显示

我们的可视化采用的是wordcloud库。这是一个‎‎单词云布局写在JavaScript。它使用HTML5画布和精灵面具，以实现近交互式的速度。将无数的繁杂的反人性的数据一图像的形式展现出来。文字越大出现的频率越大。

1.4 论文章节安排

根据软件工程专业论文要求，从系统需求分析、系统设计、系统功能、实现测试、总结与展望等几大步骤详细说明了基于公开情报的态势感知系统的设计思想和步骤。本文将实现一个基于多线程采集的分析系统。下面将以各章节的形式对本系统进行简单的介绍。

第一章绪论，简单的介绍了本系统设计的意义、阐述了大数据下情绪分析的国内外研究现状。对于本系统进行了主要功能的介绍和说明了论文章节安排的情况。

第二章相关技术内容，对使用的关键技术来介绍本系统，对数据处理的方法和步骤进行了简单的介绍。

第三章系统需求分析，该部分详细介绍了系统分析，功能性目标分析。

第四章系统设计，展示系统的设计思路

第五章系统功能模块代码实现，对本软件的功能的实现过程做了介绍,同时也详细地介绍了各个模块的实现步骤与过程。

第六章实现测试实现，对本程序进行测试并附上测试截图。

第六章总结与展望，对本系统的设计进行未来的展望，更加完善本项目。

1.5 本章小结

通过对本章节的描述，介绍了情绪分析系统研究的背景与意义，阐述了国内外的研究与对比，初步的对系统的整体设计进行了分析，同时规划了论文的整体架构和章节划分情况。

**2** **相关技术介绍**

本系统是基于对百度分析的系统，。本系统编写主要用到Python、html、css、jiagu、Snownlp、cemotion、sentiment、baiduspider、wordcloud、django，所运用到的技术主要包括三大块：数据采集、数据分析、数据可视化 ，下面将对以下技术部分一一介绍。

2.1 数据采集

情数据采集使用Django,Python的baiduspider爬虫爬取百度上的8页数据，并将数据转化成一个字符串，然后用jieba将数据切成很多中文单词并存入数组中。

**Django:**

‎Django 是一个高级 Python Web 框架，鼓励快速的开发和干净、务实的设计。是一个完美主义者的网络框架。

**BaiduSpider：**

提供简易的，人性化的API，可通过实例化BaiduSpider对象后调用。

**Jieba：**

支持四种分词模式：精确模式，试图将句子最精确地切开，适合文本分析;

全模式，把句子中所有的可以成词的词语都扫描出来， 速度非常快，但是不能解决歧义;搜索引擎模式，在精确模式，对长词再次切分，提高召回率，适合用于搜索引擎分词。paddle模式，利用PaddlePaddle深度学习框架，训练序列，实现分词。 同时支持词性标。 paddle模式使用需安装paddlepaddle-tiny。 支持繁词分词，支持自定义词典、MIT 授权协议。

2.2数据分析

本程序使用的数据分析主要是情绪分析。本程序使用了jiagu、Snownlp、cemotion、sentiment来进行数据分析。因为分析对象是中文单词数组并非句子，所以并没有多个特征，贝叶斯算法在此程序中用处不大，更多的是看各自库自带的词典准确度。

**Jiagu**：

Jiagu是深度学习自然语言处理工具，拥有知识图谱关系抽取、中文分词、 词性标注、命名实体识别、情感分析、新词发现、关键词、文本摘要、文本聚类的功能。

**Snownlp**：

用于处理中文文本文档。现在训练数据主要是买卖东西时的评价，所以对其他的一些可能效果不是很好。

**Cemotion**：

Python的基于NLP的中文情感倾向分析库。但处理速度慢！

**Sentiment：**

用node.Js的基于AFINN的情绪分析算法。

2.3数据可视化

通过wordcloud生成词云图来对进行情绪处理后的数组数据进行可视化。将各自带有情绪的词生成词云图，用频率大小来控制文字大小。使对应的关键字的好评、差评的关键字一目了然。

Wordcloud：

100%用JavaScript创建的生成词云图的库。

2.4本章小结

本章主要对情绪分析词云系统的数据采集与处理与可视化进行了详细的介绍，说明了系统可视化界面实现所运用到的技术，分析了数据采集技术使用的爬虫方法。数据分析部分分析了当前学术界中对于情绪识别的常见算法（jiagu、Snownlp、cemotion、sentiment）。数据可视化部分分析了本程序用到的可视化方法wordcloud，以及它的基本介绍。

**3 系统需求分析**

需求分析也被称为软件需求分析、系统需求分析或需求分析工程，本文对情绪分析词云系统进行分析，确保能够达到系统最终的目的与效果。本系统设计过程中主要使用Python语言和h5、css来完成相关的开发，使用baiduspider来进行数据采集，使用jieba来进行中文分词，使用jiagu等情态分析库来进行数据分析，使用wordcloud来进行词云生成，使用css等前端工具来进行数据展示，一步步来完成系统。

3.1系统需求分析

现代科学技术发展的速度越来越快，新的科技知识和信息量迅猛增加。英国学者詹姆斯·马丁统计，人类知识的倍增周期，在19世纪为50年，20世纪前半叶为10年左右，到了70年代，缩短为5年，80年代末几乎已到了每3年翻一番的程度。近年来，全世界每天发表的论文达13000～14000篇，每年登记的新专利达70万项。新理论、新材料、新工艺、新方法的不断出现，使知识老化的速度加快。据统计，一个人所掌握的知识半衰期在18世纪为80～90年，19～20世纪为30年，20世纪60年代为15年，进入80年代，缩短为5年左右。还有报告说，全球印刷信息的生产量每5年翻一番，《纽约时报》一周的信息量即相当于17世纪学者毕生所能接触到的信息量的总和。近30年来，人类生产的信息已超过过去5000年信息生产的总和。

信息缺乏管理或管理不善，信息的发布、传播失去控制，产生了大量虚假信息、无用信息，造成信息环境的污染和“信息垃圾”的产生。因为在网络上任何人都可以自由发表意见，并且发布的成本几乎可以忽略，在某种意义上，“每个人都可成为全球范围的信息制造者”。从而增加了人们利用信息的困难。

计算机病毒造成的错误信息。

信息生产者站在不同角度，用不同视角对同一事物做出不同的描述与反映，有时甚至大相径庭，使信息使用者难以抓住事物发展变化的本质与主流。

网络上的垃圾站点散布的不健康的信息。如因特网上的色情淫秽信息、教唆对计算机信息系统进行非授权访问的黑客诡计等。

量的角度来看，信息爆炸是指由于传播技术的进步以及传播环境的日渐放开，信息呈海量级甚至银河级的涌现。而信息匮乏指的是由于信道的阻塞或信源的不足而导致的信息量少，不能满足受众的信息消费需求。这个意义上的信息匮乏指的是一种绝对的匮乏。

从质的角度上讲，伴随着信息爆炸的是信息的泥沙俱下，海量的信息中鱼龙混杂，从而使得真正有价值的信息被大量垃圾信息所淹没，受众往往面对庞杂的信息茫然无措，这就造成了另一种形式的信息匮乏，相对的信息匮乏。

**不利影响：**

实际上目前的状况是：人们一方面享受着网络上丰富的信息带来的便利，另一方面也在忍受着“信息爆炸”的困扰。 “信息爆炸”已经对社会经济的发展产生了负面的影响。对世界10家跨国公司的调查表明，由于每天要处理的信息超过它们的分析能力，妨碍了他们的决策效率，甚至导致决策失误或是难以作出最佳决策。有分析说，目前收集不少信息所花费的成本已超过了信息本身的价值。

**信息爆炸引发阅读模式变革：**

给人类社会带来了负面效应和潜在危机。如何使信息资源得到有效的利用，提高信息的质量，已经成为一个世界性的亟待医治的网络顽症。所以，优化信息资源的开发、管理是使信息被有效利用的关键问题之一。据报道，美国基础软件开发商英克托米公司1月18日公布了一项研究结果：目前因特网上可编索引的网页已超过10亿，如果加上大量无法编索引的网页，因特网网页总数则更多。

在进入英克托米数据库的10亿多网页中，近87%为英语，法语网页占2.4%。在因特网上，可进入的网站有420多万个，不可进入的网站有73万多个。网站互相链接，仅与美国网络公司雅虎的链接站点就有75万个。

目前因特网上除了存在丰富的有益信息外，还存在大量色情、暴力、虚假广告等有害信息，这些垃圾信息泛滥已形成网络新公害，就像大城市中的工业垃圾和生活垃圾造成环境和生态污染一样，阻碍了用户迅速查找有用信息，需要及时处理。

西方发达国家拥有800多颗通讯卫星，控制着世界75%以上的信息和65%以上的电视节目和广播节目。西方四大通讯社每天总发稿量高达3500万字。在我国，报纸的版面一扩再扩，办了日报又办晚报、晨报，一个报业集团内往往有十多家报刊。一些广播电台和电视台纷纷办成广电集团，除了实行正点滚动播出新闻之外，还兴办新台，如经济台、交通台、娱乐台、教育台等。在沿海省会城市，一般都办有60多种报纸；地级城市还分别办有五六种报纸；各个县、市也办有报纸、广播和电视。在当代社会，电视和网络是最强大、最主要的信息“运输公司”。目前在我国的天空，除了无线和有线的广播和电视，单卫星电视就有近200个频道，除去加密电视和信号微弱的电视频道，可接收的大约也有80多个频道。这无疑是陆海空交织、声字像并举的立体化的全方位的新闻信息传播格局，其新闻信息量之巨可想而知。

大众传播媒介主要被用于娱乐所占有的百分比大得惊人。几乎全部美国商业电视、大部分畅销杂志，大部分广播，大部分商业电影，还有报纸内容中越来越大的部分，“都是以让人娱乐而不是以开导为目的的”。(施拉姆，1986)娱乐信息借助大众传媒的强力传播，遍布社会的每个角落，进人大众的每根神经，不仅大量的录像片、录音带、CD、VCD、MP3、MP4、流行歌曲、流行小说等源源不断地流向家庭，就连那些最严肃的新闻报道、政治演说、群众集会，我们也会看到娱乐信息饱蕴其中，甚至麦克风前声音甜、软、轻、柔、绵的播音员和摄像机前十分性感的俊男俏女，也无不是在竭尽全力以富有诱惑力的声音和形体向人们传播某种娱乐信息。而解决这一切的方法之一便是通过词云分析系统对数据精选提纯。对大数据的处理催生出情绪分析，以及可视化。

**情感分析**

社交媒体、电子邮件、聊天、产品评论和推荐的文本挖掘和分析已经成为几乎所有行业垂直行业研究数据模式的宝贵资源，它能够帮助企业获得更多信息、更加了解客户、预测和增强客户体验、量身定制营销活动，并协助做决策。情感分析使用机器学习算法来确定文本内容是正面或负面。手动分析客户或潜在客户产生的大量文本非常耗时，机器学习更有效，通过流分析，可以实现并提供预测。

**可视化处理**

可视化处理的方式有很多种，比如Web界面展示、Windows平台的图形界面展示、APP客户端展示、微信小程序实现、Linux平台展示等。本设计将利用网页及wordcloud和web来作为技术工具实现可视化页面。

3.3 本章小结

本章节介绍了本系统的基本需求分析，主要对大数据下情绪分析的重要性进行的说明。以及本系统使用的部分技术也有所涉及。

**4.系统设计**

1.获取数据：表单将关键字，坏词上限，好词下限值传入，利用Django,python的baiduspider库通过百度搜索的有序性获取数据，并存入元组，将元组化为字符串。

2.数据分词：利用jieba的中文分词功能将字符串进行分词并存入数组。

3.数据分析：利用循环和jiagu等情绪分析库将数组内的中文单词分为好坏两种分别存在pos和neg两个数组中。

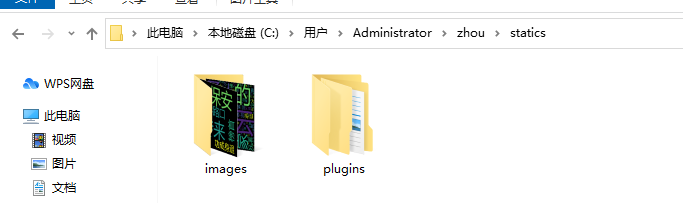
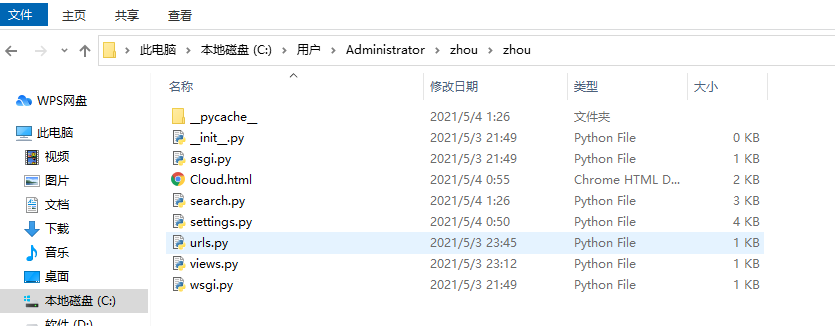
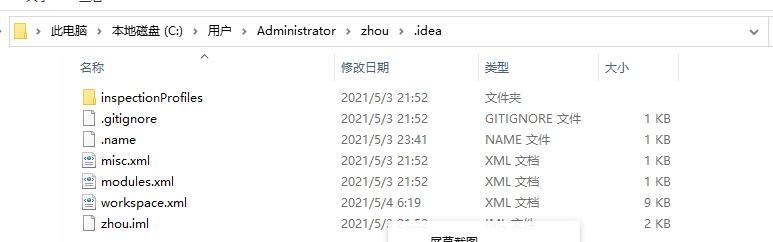
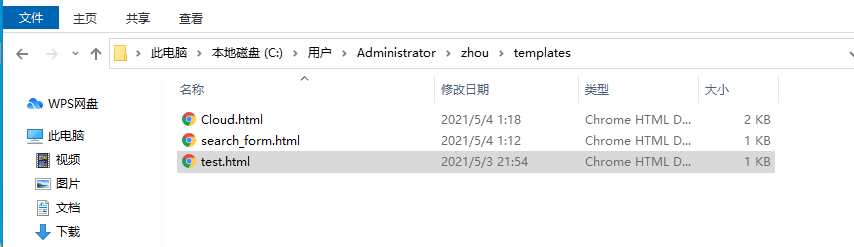
4.词云生成：将三个数组中的数据生成词云图并利用h5和css进行展示

5.系统缺陷：没有数据库导致无法并发处理；等待时间长（因为处理数据量大，且需要爬取32个baidu搜索页面，又必须等待来反爬）；因为是先对爬取数据进行jieba分词，所以情态分析分析的是词，而不是句子这就导致分析不准且实际上只用到了对应的词的字典而并没有用到对应多个特征值的预测算法---贝叶斯算法；因为太麻烦而并未布局到服务器上。

**5.代码**

5.1代码结构





5.2具体代码

在本设计中主要由数据采集模块、数据处理模块、可视化界面显示等组成。系统主要以baiduspider进行数据采集，jieba进行中文分词，情态分析算法进行数据分析，wordcloud进行数据词云可视化。

**1）解决反爬虫问题代码实现**

#多设置几个代理

ua\_list={#设置多个user-agent

'Mozilla/5.0 (Windows NT 10.0; WoW64) AppleWebKit/536.36

(KHTMT,like Gecko) Chrome/63.0.3239.132 Safair/537.36 QIHU 360SE',

}

#增加访问时间响应，模拟人的正常上网行为，两秒钟一次请求

import time

time.sleep(2)

**2）爬取8页百度数据，并合并成字符串**

spider = BaiduSpider()#创建spider对象

# 搜索网页

keyword = request.GET['q']#获取表单传过来的关键字

bad = float(request.GET['min'])#获取表单传过来的坏词最大值

good = float(request.GET['max'])#获取表单传过来的好词最小值

test\_list0 = spider.search\_web(query=keyword, pn=1)#获取第一页并生成对应元组

test\_list1 = spider.search\_web(query=keyword, pn=2)#time属性可加本月，本周，本年

test\_list2 = spider.search\_web(query=keyword, pn=3)

test\_list3 = spider.search\_web(query=keyword, pn=4)

test\_list4 = spider.search\_web(query=keyword, pn=5)

test\_list5 = spider.search\_web(query=keyword, pn=6)

test\_list6 = spider.search\_web(query=keyword, pn=7)

test\_list7 = spider.search\_web(query=keyword, pn=8)

txt\_j0 = jieba.lcut(test\_list0.\_\_str\_\_())#将元组强转成字符串并用jieba进行分词成数组

txt\_j1 = jieba.lcut(test\_list1.\_\_str\_\_())

txt\_j2 = jieba.lcut(test\_list2.\_\_str\_\_())

txt\_j3 = jieba.lcut(test\_list3.\_\_str\_\_())

txt\_j4 = jieba.lcut(test\_list4.\_\_str\_\_())

txt\_j5 = jieba.lcut(test\_list5.\_\_str\_\_())

txt\_j6 = jieba.lcut(test\_list6.\_\_str\_\_())

txt\_j7 = jieba.lcut(test\_list7.\_\_str\_\_())

txt\_j = txt\_j0+txt\_j1+txt\_j2+txt\_j3+txt\_j4+txt\_j5+txt\_j6+txt\_j7

**3）snownlp情态分析核心代码实现**

positivelist = []#积极数组

negativelist = []#消极数组

for each in txt\_j:#遍历字符串

each\_word=snownlp.SnowNLP(each)#对each进行情态分词

feeling=each\_word.sentiments#情态评分

if feeling>good:#good是表单传过来的好评最小值

positivelist.append(each)#each加入pos数组

elif feeling<bad:#bad是表单传过来差评最大值

negativelist.append(each)#加入neg数组

else:

pass

positive = " ".join(positivelist)#转格式

negative = " ".join(negativelist)#转格式

test\_str = " ".join(txt\_j)

**4）jiagu核心代码实现**

positivelist\_jiagu\_month = []

negativelist\_jiagu\_month = []

for each in txt\_week\_j:

sentiment = jiagu.sentiment(each)

if sentiment[0] == 'positive' and sentiment[1] > good:

positivelist\_jiagu\_month.append(each)

elif sentiment[0] == 'negative' and sentiment[1] > (1-bad):

negativelist\_jiagu\_month.append(each)

else:

pass

positive\_jiagu\_month = " ".join(positivelist\_jiagu\_month)

negative\_jiagu\_month = " ".join(negativelist\_jiagu\_month)

**5）sentiment核心代码实现**

positivelist\_senti = []#积极数组

negativelist\_senti = []#消极数组

senti = Sentiment()

for each in txt\_j:#txt\_j是分词以后的字符串

sentiment = senti.sentiment\_count(each)#情绪分析

if sentiment['pos'] > sentiment['neg']:#如果积极的元组值大于消极的元组值

positivelist\_senti.append(each)#单词放入积极数组

elif sentiment['pos'] < sentiment['neg']:

negativelist\_senti.append(each)#单词放入消极数组

else:

pass

positive\_senti = " ".join(positivelist\_senti)#转格式

negative\_senti = " ".join(negativelist\_senti)#转格式

**7）词云生成代码**

w=wordcloud.WordCloud(width=1000,height=1000,font\_path='msyh.ttc',stopwords=['time','a','b','c','d','e','f','g','h','i','j','k','l','n','m','o','p','q','r','s','t','u','v','w','x','y','z','www','baidu','com','http','url','type','title','link','result','des','result','Copyright','None','origin'])#创建词云对象并声明宽、高、字体、停用词

w1=wordcloud.WordCloud(width=1000,height=1000,font\_path='msyh.ttc',stopwords=['time','a','b','c','d','e','f','g','h','i','j','k','l','n','m','o','p','q','r','s','t','u','v','w','x','y','z','www','baidu','com','http','url','type','title','link','result','des','result','Copyright','None','origin'])

w2=wordcloud.WordCloud(width=1000,height=1000,font\_path='msyh.ttc',stopwords=['time','a','b','c','d','e','f','g','h','i','j','k','l','n','m','o','p','q','r','s','t','u','v','w','x','y','z','related','www','baidu','com','http','url','type','title','link','result','des','result','Copyright','None','origin'])

w.generate(test\_str)#用字符串生成词云图

w1.generate(positive)

w2.generate(negative)

w1.to\_file('statics/images/positive.png')#将词云图保存方便调用

w2.to\_file('statics/images/negative.png')

w.to\_file('statics/images/test.png')

**6）搜索表单代码**

<form name="login-form" class="login-form" action="/search/" method="get">

<div class="header">

<h1>词云搜索</h1>

<span>坏词~好词(0.0~1.0)</span>

</div>

<div class="content">

<input name="q" type="text" class="input search" placeholder="搜索词" />

<input name="min" type="text" class="input" placeholder="坏词<" />

<input name="max" type="text" class="input" placeholder="好词>" />

</div>

<div class="footer">

<input type="submit" value="搜索" class="button" />

</div>

</form>

**7）search过渡核心程序**

message = '你搜索的内容为: ' + request.GET['q'] + '<a href="Cloud">观看词云图</a>'+'，坏词：0~'+request.GET['min']+',好词：'+request.GET['max']+'~1.0'

else:

message = '你提交了空表单'

return HttpResponse(message)

**8）cloud可视化一个图形的代码范例**

<div class="responsive">

<div class="img">

<a target="\_blank" href="{% static "images/test.png" %}">

{% load static %}

<img src="{% static "images/test.png" %}" alt="所有评论">

</a>

<div class="desc">所有评论</div>

</div>

</div>

。。。。。。

1. **实现消极词云显红色积极显蓝色**

import imageio

zhezhao = imageio.imread('1.png')

。。。。。

w1=wordcloud.WordCloud(font\_path='msyh.ttc', mask=zhezhao, \

width=1000, height=1000, background\_color='black', mode='RGBA',

color\_func=lambda \*args, \*\*kwargs: "blue",stopwords=['time','a','b','c','d','e','f','g','h','i','j','k','l','n','m','o','p','q','r','s','t','u','v','w','x','y','z','www','baidu','com','http','url','type','title','link','result','des','result','Copyright','None','origin'])

w2=wordcloud.WordCloud(font\_path='msyh.ttc', mask=zhezhao, \

width=1000, height=1000, background\_color='black', mode='RGBA',

color\_func=lambda \*args, \*\*kwargs: "red",stopwords=['time','a','b','c','d','e','f','g','h','i','j','k','l','n','m','o','p','q','r','s','t','u','v','w','x','y','z','related','www','baidu','com','http','url','type','title','link','result','des','result','Copyright','None','origin'])

w1.generate(positive)

w2.generate(negative)

5.4 本章小结

本章介绍了本系统的基本实现。展示了系统每个功能的部分核心代码。

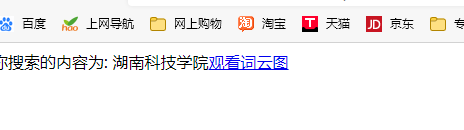
**6.测试**

**粗略实现截图：**

搜索界面：



跳转界面：



计算结果：



**7. 总结与展望**

未来的数据情态分析，可视化趋势已成为了必然，这样的趋势只有创新才能走的更远。在越来越多的企业、政府等的应用场景应用也会普及，根据实时的监控数据，把最新的数据展现在大屏幕上，清楚的看到自己想要看到的数据，同时根据这些数据做出决策的调整。

世界正变得越来越数字化，大数据正在以这种或那种方式影响着每个人的生活。我们在日常生活中所做的一切都会留下数字痕迹(或者数据)，也就是大数据，我们可以通过数据可视化工具和情态分析工具利用和分析这些数据来让我们的生活更加美好。在大数据时代不会使用大数据必将落后，落后就会挨打，一如这个世界不进则退。我们应该拥抱大数据、拥抱数据可视化、拥抱情态分析等大数据分析工具。

在市场上面的数据可视化工具、情态分析工具逐渐增多的情况下，不管使用什么样的工具，能够满足自己需求的工具就是好工具，最后希望可视化工具和情态分析工具能够更多的帮助到你。

**8. 项目链接**

**项目GitHub地址：**<https://github.com/neverdied/Word-cloud-search>

**个人主页地址：**[周の素像 (zhouxinwen.top)](https://zhouxinwen.top/)

**个人github主页：https://github.com/neverdied**