结巴分词和NLTK----一套中文文本分析的组合拳

Hellooooooworld 关注
0.2 2017.05.08 21:16:44 字数 749 阅读 17,145

> 汉语是世界上最难学的语言！

有人说汉语难学难懂，那么对中文文本的分析也就相对于英文文本来说，更加困难！
在学习的过程中，我最先接触的是NLTK和jieba这两个python的自然语言包，前者，我主要是对分词后的数据进行分析；而后者，我主要用于对文章进行分词！

part.1 包的安装

```
pip install nltk
pip install jieba
```

一条文本分析的漫漫长路就此开始！

part.2 怎么使用

从这里开始，我根据着自己写的类来对这两个包进行一些简单的讲解。

2.1 引用

```
# coding=utf-8
# -*- coding: cp936 -*-
import jieba
import jieba.posseg as pseg
import codecs
import re
import os
import time
import string
from nltk.probability import FreqDist
open=codecs.open
```

2.2 自定义词典和停用词的引入

自定义词典是我们在分词的时候避免把我们需要的词组分成小词而导入的，而停用词，则是我们在分词过程中，将对我们分词过程中的干扰词排除在外的词典。

```
#jieba 分词可以将我们的自定义词典导入，格式“词”“词性”“词频”
jieba.load_userdict('data/userdict.txt')

#定义一个keyword类
class keyword(object):
    def Chinese_Stopwords(self):          #导入停用词库
        stopword=[]
        cfp=open('data/stopWord.txt','r+','utf-8')   #停用词的txt文件
        for line in cfp:
            for word in line.split():
                stopword.append(word)
        cfp.close()
        return stopword
```

Hellooooooworld 关注
总资产1 (约0.13元)

kvm+bond+bridge 实现多网卡的网桥搭建
阅读 154

[linux] sz 与 rz
阅读 33

推荐阅读

tensorflow常用函数
阅读 1,588

Modeling Task Relationships in Multi-task Learning with Multi-gat...
阅读 2,023

Python的开源人脸识别库：离线识别率高达99.38%
阅读 7,552

Pytorch-激活函数
阅读 622

推荐系统排序算法--DIN模型
阅读 2,323

行后续的操作。

```
    def Word_cut_list(self,word_str):
        #利用正则表达式去掉一些一些标点符号之类的符号。
        word_str = re.sub(r'\s+', ' ', word_str)  # trans 多空格 to空格
        word_str = re.sub(r'\n+', ' ', word_str)  # trans 换行 to空格
        word_str = re.sub(r'\t+', ' ', word_str)  # trans Tab to空格
        word_str = re.sub("[\s+\.\!\/_,$%^*(+\"\']+|[+——！，”。《》，。：“？、~@#￥%……&*（）12345
                          decode("utf8"), "".decode("utf8"), word_str)

        wordlist = list(jieba.cut(word_str))#jieba.cut  把字符串切割成词并添加至一个列表
        wordlist_N = []
        chinese_stopwords=self.Chinese_Stopwords()
        for word in wordlist:
            if word not in chinese_stopwords:#词语的清洗：去停用词
                if word != '\r\n'  and word!=' ' and word != '\u3000'.decode('unicode_escape') \
                        and word!='\xa0'.decode('unicode_escape'):#词语的清洗：去全角空格
                    wordlist_N.append(word)
        return wordlist_N
```

什么叫挑选呢?
其实在我们进行中文文本的分析时，不是每个词都有用的。那什么样的词就能表述出文章意思呢?
比如：名词!
那怎么把名词提取出来呢? 👇👇👇👇👇

```
    def Word_pseg(self,word_str):  # 名词提取函数
        words = pseg.cut(word_str)
        word_list = []
        for wds in words:
            # 筛选自定义词典中的词，和各类名词，自定义词库的词在没设置词性的情况下默认为x词性，即词的fla
            if wds.flag == 'x' and wds.word != ' ' and wds.word != 'ns' \
                    or re.match(r'^n', wds.flag) != None \
                            and re.match(r'^nr', wds.flag) == None:
                word_list.append(wds.word)
        return word_list
```

2.4 排序和运行

先前，我们对分词和分词后的挑选进行了一定的分析了解，那么怎么把我们得到的这个此列表进行分析呢？简单的就是先统计词频，这样一分析，有词频，我们自然而然就想到了排序。

```
    def sort_item(self,item):#排序函数，正序排序
        vocab=[]
        for k,v in item:
            vocab.append((k,v))
        List=list(sorted(vocab,key=lambda v:v[1],reverse=1))
        return List

    def Run(self):
        Apage=open(self.filename,'r+','utf-8')
        Word=Apage.read()                       #先读取整篇文章
        Wordp=self.Word_pseg(Word)              #对整篇文章进行词性的挑选
        New_str=''.join(Wordp)
        Wordlist=self.Word_cut_list(New_str)    #对挑选后的文章进行分词
        Apage.close()
        return  Wordlist

    def __init__(self, filename):
        self.filename = filename

```

2.5 main函数的读取分析

对我们的研究很是偏颇，那么我就使用百分比来输出关键词。

```
if __name__=='__main__':
    b_path = 'data/all'
    a_path = 'data/Result'
    roots = os.listdir(b_path)
    alltime_s = time.time()
    for filename in roots:
        starttime = time.time()
        kw = keyword(b_path + '/' + filename)
        wl = kw.Run()
        fdist = FreqDist(wl)
        Sum = len(wl)
        pre = 0
        fn = open(a_path + '/' + filename, 'w+', 'utf-8')
        fn.write('sum:' + str(Sum) + '\r\n')
        for (s, n) in kw.sort_item(fdist.items()):
            fn.write(s + str(float(n) / Sum)+"      " +str(n)+ '\r\n')
            pre = pre + float(n) / Sum
            if pre > 0.5:
                fn.write(str(pre))
                fn.close()
                break
        endtime = time.time()
        print filename + '       完成时间：' + str(endtime - starttime)

    print "总用时：" + str(time.time() - alltime_s)
```

part.3 简单总结

在我们进行中文文本分析的路上，我目前做的只是冰山一角，只是简单地进行一些文本的分词统计，这还是只是个开始。

Nltk gensim nltk

推荐阅读

更多精彩内容 >

NLP常用专业术语

常用概念： 自然语言处理（NLP） 数据挖掘 推荐算法 用户画像 知识图谱 信息检索 文本分类 常用技术： 词级别...

御风之星 阅读 5,040 评论 1 赞 24

中文分词项目总结

1）ICTCLAS 最早的中文开源分词项目之一，由中科院计算所的张华平、刘群所开发，采用C/C++编写，算法基于《...

stonelin3935 阅读 4,443 评论 1 赞 14

中文分词算法总结

转载请注明：终小南 » 中文分词算法总结 什么是中文分词众所周知，英文是以 词为单位的，词和词之间是靠空格隔开，而...

kirai 阅读 7,384 评论 3 赞 21

Python 网页爬虫 & 文本处理 & 科学计算 & 机器学习 & 数据挖掘兵器谱

为了自己以后应用的方便，于是将这篇文章转载到这里。Python 网页爬虫 & 文本处理 & 科学计算 & 机器学习...

tianmh 阅读 65,185 评论 0 赞 64

福州新生儿医保/上户口/报销攻略

前言 娃出生一个多月，深深地被娃上户口，医保，报销等事虐的没脾气了。有感于此，特意分享出来，希望和我们一样的新手爸...

黑羽肃霜 阅读 26,046 评论 1 赞 10