python 结巴分词(jieba)

简介

"结巴"中文分词：做最好的Python中文分词组件 "Jieba"

源码下载的地址：https://github.com/fxsjy/jieba 演示地址：http://jiebademo.ap01.aws.af.cm/

特点

支持三种分词模式

- 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，在精确模式的基础上，对长词再次切分，提高召回率，适合用于搜索引擎分词。

注：支持繁体分词 支持自定义词典

安装

Python 2.x 下的安装

- 全自动安装：easy_install jieba 或者 pip install jieba
- 半自动安装：先下载http://pypi.python.org/pypi/jieba/ ，解压后运行python setup.py install
- 手动安装：将jieba目录放置于当前目录或者site-packages目录

通过import jieba 来引用

Python 3.x 下的安装

目前master分支是只支持Python2.x 的 Python3.x 版本的分支也已经基本可用：
https://github.com/fxsjy/jieba/tree/jieba3k

```
git clone https://github.com/fxsjy/jieba.git
```

```
git checkout jieba3k
python setup.py install
```

算法实现

基于Trie树结构实现高效的词图扫描，生成句子中汉字所有可能成词情况所构成的有向无环图（DAG) 采用了动态规划查找最大概率路径, 找出基于词频的最大切分组合 对于未登录词，采用了基于汉字成词能力的HMM模型，使用了Viterbi算法

功能

分词

jieba.cut方法接受两个输入参数: 1) 第一个参数为需要分词的字符串 2）cut_all参数用来控制是否采用全模式

jieba.cut_for_search方法接受一个参数：需要分词的字符串,该方法适合用于搜索引擎构建倒排索引的分词，粒度比较细

注意：待分词的字符串可以是gbk字符串、utf-8字符串或者unicode

jieba.cut以及jieba.cut_for_search返回的结构都是一个可迭代的generator，可以使用for循环来获得分词后得到的每一个词语(unicode)，也可以用list(jieba.cut(...))转化为list

代码示例(分词)

```
#encoding=utf-8
import jieba
seg_list = jieba.cut("我来到北京清华大学", cut_all=True)
print "Full Mode:", "/ ".join(seg_list)  # 全模式
seg_list = jieba.cut("我来到北京清华大学", cut_all=False)
print "Default Mode:", "/ ".join(seg_list)  # 精确模式
seg_list = jieba.cut("他来到了网易杭研大厦")  # 默认是精确模式
print ", ".join(seg_list)
seg_list = jieba.cut_for_search("小明硕士毕业于中国科学院计算所，后在日本京都大学深造")  # 搜
print ", ".join(seg_list)
```

Output:

```
【全模式】: 我/ 来到/ 北京/ 清华/ 清华大学/ 华大/ 大学
【精确模式】: 我/ 来到/ 北京/ 清华大学
【新词识别】: 他，来到，了，网易，杭研，大厦    (此处，“杭研”并没有在词典中，但是也被Viterbi算
【搜索引擎模式】:  小明，硕士，毕业，于，中国，科学，学院，科学院，中国科学院，计算，计算所，后
```

添加自定义词典

开发者可以指定自己自定义的词典，以便包含jieba词库里没有的词。虽然jieba有新词识别能力，但是自行添加新词可以保证更高的正确率

用法：

```
jieba.load_userdict(file_name) # file_name为自定义词典的路径
```

词典格式和dict.txt一样，一个词占一行；每一行分三部分，一部分为词语，另一部分为词频，最后为词性（可省略），用空格隔开

范例：

自定义词典：

```
云计算 5
李小福 2 nr
创新办 3 i
easy_install 3 eng
好用 300
韩玉赏鉴 3 nz
```

用法示例：

```
#encoding=utf-8
import sys
sys.path.append("../")
import jieba
jieba.load_userdict("userdict.txt")
import jieba.posseg as pseg

test_sent = "李小福是创新办主任也是云计算方面的专家;"
test_sent += "例如我输入一个带“韩玉赏鉴”的标题，在自定义词库中也增加了此词为N类型"
words = jieba.cut(test_sent)
for w in words:
print w
```

```
result = pseg.cut(test_sent)

for w in result:
print w.word, "/", w.flag, ", ",

print "\n========"

terms = jieba.cut('easy_install is great')
for t in terms:
print t
print '-------------------------'
terms = jieba.cut('python 的正则表达式是好用的')
for t in terms:
print t
```

output：

```
之前： 李小福 / 是 / 创新 / 办 / 主任 / 也 / 是 / 云 / 计算 / 方面 / 的 / 专家 /
加载自定义词库后： 李小福 / 是 / 创新办 / 主任 / 也 / 是 / 云计算 / 方面 / 的 / 专家 /
"通过用户自定义词典来增强歧义纠错能力" --- https://github.com/fxsjy/jieba/issues/14
```

关键词提取

```
jieba.analyse.extract_tags(sentence,topK) #需要先import jieba.analyse
```

说明

setence为待提取的文本

topK为返回几个TF/IDF权重最大的关键词，默认值为20

代码示例 （关键词提取）

```
import sys
sys.path.append('../')

import jieba
import jieba.analyse
from optparse import OptionParser

USAGE = "usage: python extract_tags.py [file name] -k [top k]"

parser = OptionParser(USAGE)
```

```
parser.add_option("-k", dest="topK")
opt, args = parser.parse_args()


if len(args) < 1:
    print USAGE
    sys.exit(1)

file_name = args[0]

if opt.topK is None:
    topK = 10
else:
    topK = int(opt.topK)

content = open(file_name, 'rb').read()

tags = jieba.analyse.extract_tags(content, topK=topK)

print ",".join(tags)
```

词性标注

标注句子分词后每个词的词性，采用和ictclas兼容的标记法

用法示例

```
>>> import jieba.posseg as pseg
>>> words = pseg.cut("我爱北京天安门")
>>> for w in words:
...    print w.word, w.flag
...
我 r
爱 v
北京 ns
天安门 ns
```

并行分词

原理

将目标文本按行分隔后，把各行文本分配到多个python进程并行分词，然后归并结果，从而获得分词速度的可观提升

基于python自带的multiprocessing模块，目前暂不支持windows

用法

```
jieba.enable_parallel(4) # 开启并行分词模式，参数为并行进程数
jieba.disable_parallel() # 关闭并行分词模式
```

例子

```
import urllib2
import sys,time
import sys
sys.path.append("../../")
import jieba
jieba.enable_parallel(4)

url = sys.argv[1]
content = open(url,"rb").read()
t1 = time.time()
words = list(jieba.cut(content))

t2 = time.time()
tm_cost = t2-t1

log_f = open("1.log","wb")
for w in words:
print >> log_f, w.encode("utf-8"), "/" ,

print 'speed' , len(content)/tm_cost, " bytes/second"
```

实验结果：在4核3.4GHz Linux机器上，对金庸全集进行精确分词，获得了1MB/s的速度，是单进程版的3.3倍。

其他词典

- 占用内存较小的词典文件 https://github.com/fxsjy/jieba/raw/master/extra_dict/dict.txt.small
- 支持繁体分词更好的词典文件 https://github.com/fxsjy/jieba/raw/master/extra_dict/dict.txt.big
- 下载你所需要的词典，然后覆盖jieba/dict.txt 即可或者用jieba.set_dictionary('data/dict.txt.big')

模块初始化机制的改变:lazy load （从0.28版本开始）

jieba采用延迟加载，"import jieba"不会立即触发词典的加载，一旦有必要才开始加载词典构建trie。如果你想手工初始jieba，也可以手动初始化。

```
import jieba
jieba.initialize()  # 手动初始化（可选）
```

在0.28之前的版本是不能指定主词典的路径的，有了延迟加载机制后，你可以改变主词典的路径:

```
jieba.set_dictionary('data/dict.txt.big')
```

例子

```
#encoding=utf-8
import sys
sys.path.append("../")
import jieba

def cuttest(test_sent):
result = jieba.cut(test_sent)
print " ".join(result)

def testcase():
cuttest("这是一个伸手不见五指的黑夜。我叫孙悟空，我爱北京，我爱Python和C++。")
cuttest("我不喜欢日本和服。")
cuttest("雷猴回归人间。")
cuttest("工信处女干事每月经过下属科室都要亲口交代24口交换机等技术性器件的安装工作")
cuttest("我需要廉租房")
cuttest("永和服装饰品有限公司")
cuttest("我爱北京天安门")
cuttest("abc")
cuttest("隐马尔可夫")
cuttest("雷猴是个好网站")

if __name__ == "__main__":
testcase()
jieba.set_dictionary("foobar.txt")
print "================================"
testcase()
```