Part1、使用 Jieba 分词包进行分词训练

题目分析：

对下述一段文字进行四种模式的分词（全模式，精确模式，搜索引擎模式，以及用户自定义模式）：

```
Minecraft所呈现的世界并不是华丽的画面与特效，而是注重在游戏性上面。玩家可以在游戏中的三维空间里创造和破坏游戏里的方块，甚至在多人服务器与单人世界中体验不同的游戏模式，打造精妙绝伦的建筑物，创造物和艺术品。Minecraft着重于让玩家去探索、交互、并且改变一个由一立方米大小的方块动态生成的地图。除了方块以外，环境单体还包括植物、生物与物品。游戏里的各种活动包括采集矿石、与敌对生物战斗、合成新的方块与收集各种在游戏中找到的资源的工具。游戏中的无限制模式让玩家在各种多人游戏服务器或他们的单人模式中进行创造建筑物、作品与艺术创作。其他功能包括逻辑运算与远程动作的红石电路、矿车及轨道，以及称之为"下界"的维度。最终，可以前往一个叫做"末地"的维度冒险，并击败末影龙。
```

并实现输出词性的功能。

实验具体步骤截图：

首先说明使用 utf-8 编码格式，并引入 jieba 库。

```
#-*- coding:utf-8 -*-
import jieba
```

之后把这段文字放在一个变量里：

```
sent="Minecraft所呈现的世界并不是华丽的画面与特效，而是注重在游戏性上面。玩家可以在游戏中的三维空间里创造和破坏
```

1) 全模式

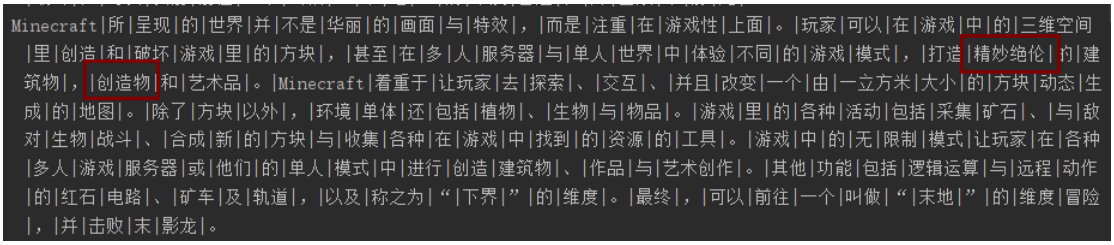
```
wordlist=jieba.cut(sent,cut_all=True)
print("|".join(wordlist))
```

```
Minecraft|所|呈现|的|世界|并|不是|华丽|的|画面|与|特效|，|而是|注重|在|游戏|游戏性|上面|。|玩家|可以|在|游戏|中|的|三维
|三维空间|维空间|空间|里|创造|和|破坏|游戏|里|的|方块|，|甚至|在|多|人|服务|服务器|务器|与|单人|人世|世界|中体|体验|不
同|的|游戏|模式|，|打造|精妙|精妙绝伦|妙绝|绝伦|的|建筑|建筑物|，|创造|创造物|造物|和|艺术|艺术品|。|Minecraft|着重|着
重于|重于|让玩家|玩家|去|探索|、|交互|、|并且|改变|一个|由|一立方米|立方|立方米|大小|的|方块|动态|生成|的|地图|。|除了
|方块|以外|，|环境|单体|还|包括|植物|、|生物|与|物品|。|游戏|里|的|各种|活动|包括|采集|矿石|、|与|敌对|对生|生物|生物战
|战斗|、|合成|新|的|方块|与|收集|各种|在|游戏|中|找到|的|资源|的|工具|。|游戏|中|的|无限|限制|制模|模式|让玩家|玩家|在
|各种|多|人|游戏|戏服|服务|服务器|务器|或|他们|的|单人|模式|中|进行|创造|建筑|建筑物|、|作品|与|艺术|艺术创作|创作|。
|其他|功能|包括|逻辑|逻辑运算|运算|与|远程|动作|的|红|石|电路|、|矿车|及|轨道|，|以及|称之为|之为|“|下界|”|的|维度|。
|最终|，|可以|以前|前往|一个|叫做|“|末|地|”|的|维度|冒险|，|并|击败|末|影|龙|。
```

可以看到把句子中所有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。这种全模式，会根据字典，将所有出现的字词全部匹配划分，所以会出现重复。

2) 精确模式

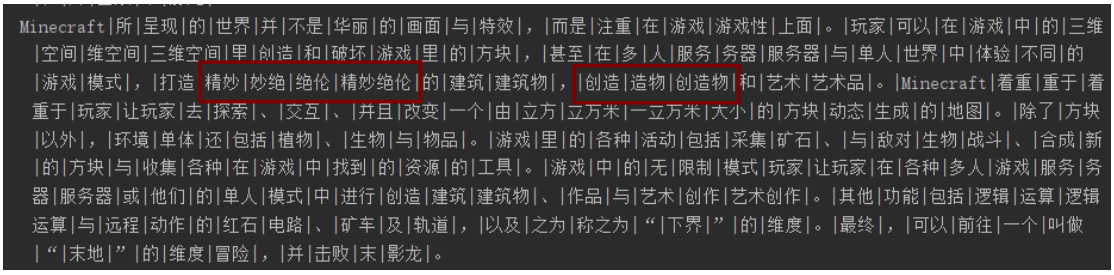
```
wordlist=jieba.cut(sent)#cut_all=Flase,和不写cut_all效果一样
print("|".join(wordlist))
```



可以看到试图将句子最精确地切开，适合文本分析。

3) 搜索引擎模式

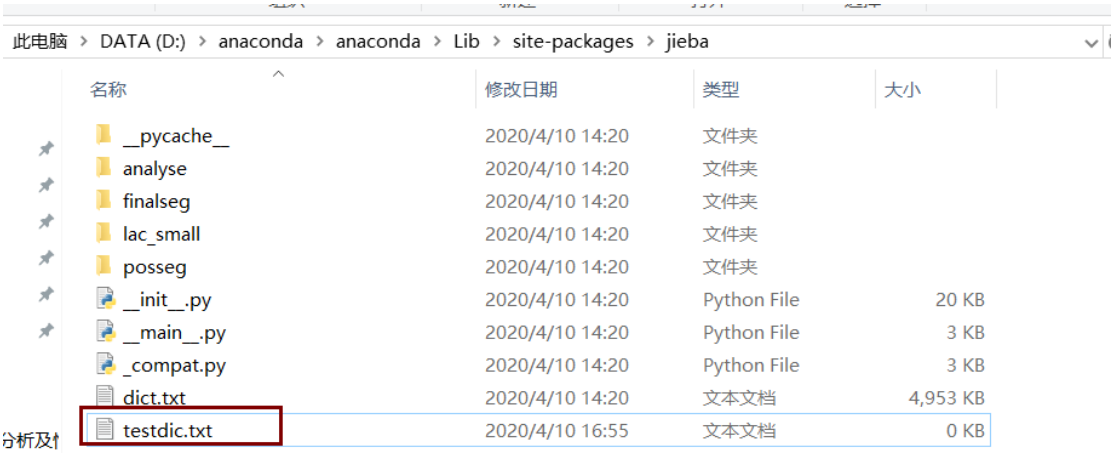
```
wordlist=jieba.cut_for_search(sent)
print(' |'.join(wordlist))
```



可以看到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。

4) 用户自定义模式

在安装包目录下，添加一个新的 txt 文件，放用户自定义字典：



在里面添加自定义的字典，格式为 1. 词　2. 数字（代表词频，越高越容易匹配到）　3. 词性　：

接下来写代码使用自定义的字典：

```
#用户自定义模式
jieba.load_userdict('D:\\anaconda\\anaconda\\Lib\\site-packages\\jieba\\testdic.txt')
wordlist=jieba.cut(sent)#cut_all=Flase
print("|".join(wordlist))
```

上面的几种模式都是把“末影龙”这种专业名词分开的，所以我们自己设计字典，优先级设置高一点，得到下面结果：

```
Minecraft|所|呈现|的|世界|并|不是|华丽|的|画面|与|特效|，|而是|注重|在|游戏性|上面|。|玩家|可以|在|游戏|中|的|三维空间|里|创造|和|破坏|游戏|里|的|方块|，|甚至|在|多|人|服务器|与|单人|世界|中|体验|不同|的|游戏|模式|，|打造|精妙绝伦|的|建筑物|，|创造物|和|艺术品|。|Minecraft|着重于|让玩家|去|探索|、|交互|、|并且|改变|一个|由|一立方米|大小|的|方块|动态|生成|的|地图|。|除了|方块|以外|，|环境|单体|还|包括|植物|、|生物|与|物品|。|游戏|里|的|各种|活动|包括|采集|矿石|、|与|敌对|生物|战斗|、|合成|新|的|方块|与|收集|各种|在|游戏|中|找到|的|资源|的|工具|。|游戏|中|的|无|限制|模式|让玩家|在|各种|多|人|游戏|服务器|或|他们|的|单人|模式|中|进行|创造|建筑物|、|作品|与|艺术创作|。|其他|功能|包括|逻辑运算|与|远程|动作|的|红石|电路|、|矿车|及|轨道|，|以及|称之为|“|下界|”|的|维度|。|最终|，|可以|前往|一个|叫做|“|末地|”|的|维度|冒险|，|并|击败|末影龙|。
```

可以看到成功把“末影龙”分为一个词！

5) 输出词性：

```
import jieba.posseg as pseg
```

```
#输出词性
words = pseg.cut("我是mc骨灰级玩家")
for word, flag in words:
    print('%s %s' % (word, flag))
```

```
我 r
是 v
mc eng
骨灰级 n
玩家 n
```

Part2、使用 Snownlp 进行情感极性分析

首先引入库并指明编码方式：

```
#-*- coding:utf-8 -*-
from snownlp import SnowNLP
```

之后使用 snowNLP 中的 sentiments 方法对句子进行情感分析。

```
print("辣鸡商家，卖的商品太差了：",SnowNLP(u'辣鸡商家，卖的商品太差了').sentiments)
print("你不要再这样了：",SnowNLP(u'你不要再这样了').sentiments)
print("这种行为真的很可恶：",SnowNLP(u'这种行为真的很可恶').sentiments)
print("我觉得你和3岁小孩一样幼稚：",SnowNLP(u'我觉得你和3岁小孩一样幼稚').sentiments)
print("自己的事情自己做去，不要打扰我：",SnowNLP(u'自己的事情自己做去，不要打扰我').sentiments)
print("你今天化的妆好漂亮啊：",SnowNLP(u'你今天化的妆好漂亮啊').sentiments)
print("出淤泥而不染,濯清涟而不妖：",SnowNLP(u'出淤泥而不染,濯清涟而不妖').sentiments)
print("你变瘦了：",SnowNLP(u'你变瘦了').sentiments)
print("我很强，我知道：",SnowNLP(u'我很强，我知道').sentiments)
print("干得漂亮：",SnowNLP(u'干得漂亮').sentiments)
```

结果如下：

```
辣鸡商家，卖的商品太差了： 0.009092224727774045
你不要再这样了： 0.41272051996285974
这种行为真的很可恶： 0.2135348298386156
我觉得你和3岁小孩一样幼稚： 0.2798686165949472
自己的事情自己做去，不要打扰我： 0.5422432793184044
你今天化的妆好漂亮啊： 0.953894129557326
出淤泥而不染,濯清涟而不妖： 0.964498270012059
你变瘦了： 0.9464603887711137
我很强，我知道： 0.7947809159406379
干得漂亮： 0.7312644688240393
```

后面分数值表示情感为 positive 的概率，越接近 1 表示越积极，越接近 0 表示越消极。