**信 息 内 容 安 全 实 验 报 告**

|  |  |
| --- | --- |
| 实验项目名称： | 人民名义图谱分析 |
| 班级： | SC011701 |
| 学号： | 2017302238 |
| 姓名： | 郑启睿 |
| 指导教师： | 杨黎斌 |
| 实验时间： | 2020.4.11 |

一、实验名称：人民名义图谱分析

二、实验内容：

利用以人民民义的剧情梗概为材料，建立剧情人物关系图谱

三 、实验要求：

1.用Python对剧情梗概进行解析，并得到人物节点；  
2.对人物节点利用相关工具进行图谱分析。  
3.有兴趣同学可以对红楼梦或权利游戏等书籍进行任务图谱分析；

四、实验背景

本次实验的对象可以是任意一本小说，从网络上下载电子版，将文本中所有的人名摘取下来，做成自定义词典，以备后来分词时使用。选取的小说人物越多越好，因为人物众多，关系错综复杂，对于喜欢的读者来说，通读全文耗时较长，所以用计算机的方式分析全文，找到人物间的关联。通过共现分析研究文章中人物与人物之间的联系，进一步地挖掘出不同人物的社交圈。使用共现的方法得出的人物关系图可以帮助读者理清著作中的复杂人物关联，更好地阅读和理解作品；用共现分析方法研究人物关系，由可视化软件呈现出的人物关系图更加直观，它抛开人们的主观判断，以数据的大小为分析依据来说明人物之间的亲密程度。

五、理论基础

本实验基于共现分析实现人民名义人物关系的图谱化。“共现”指文献的特征项描述的信息共同出现的现象，这里的特征项包括文献的外部和内部特征，如题名、作者、关键词、机构等。而“共现分析”是对共现现象的定量研究，以揭示信息的内容关联和特征项所隐含的知识。

六、实验过程

（1）流程概述

1、在网上查询《人民的名义》的一些主要人物，做一个人物字典

2、基于共现原理寻找人物关系

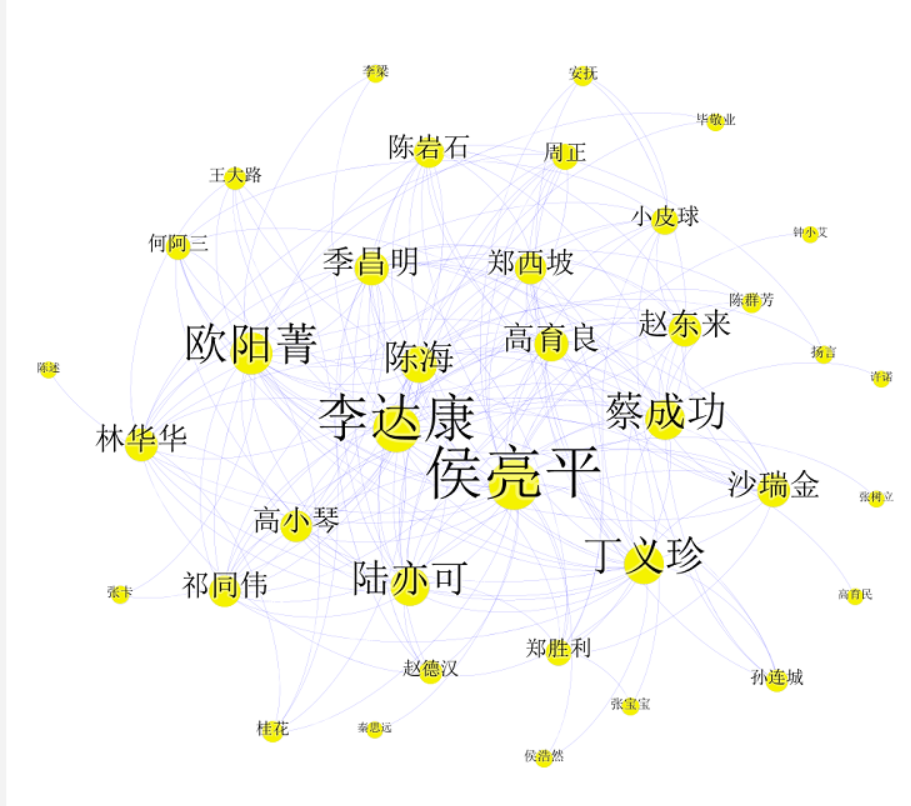
3、使用软件Gephi绘制关系图，将关系可视化

（2）处理过程中的一些问题

一些词汇如“林城”、“白云”、“吕州”等，它们的属性被归到了“nr”nr”，而且出现次数较多，和一些人物共现次数较多，如果不做处理的话，它们将会出现在绘制的关系图中。在这里可以将这些词作为停用词梳理。

一些人物之间常常会以亲密的词汇称呼对方，而且出现次数非常多，如果将这些词汇忽略，关系刻画可能会出现很大的偏差，如陈海和侯亮平之间的“猴子”，高育良和侯亮平、祁同伟之前的师生称呼。所以在分词之后，我们可以判断一下如果出现了这些亲密称呼，将其替代为他的姓名。

七、结果分析



实验中的不足：

由于不熟悉gephi的应用，没有过滤掉一些并不重要的人物节点，导致效果图比较复杂，整体设计也不够美观。

八、实验心得

学习了共现原理，初步掌握了gephi的应用。

实验源码：

# -\*- encoding:utf-8 -\*-

import jieba

import jieba.posseg as pseg

import codecs

import csv

stopwords=['吕州','林城','银行卡','明白','白云','嗡嗡嘤嘤',

'阴云密布','雷声','陈大','谢谢您','安置费','任重道远',

'孤鹰岭','阿庆嫂','岳飞','师生','养老院','段子','老总']

replace\_words={'师母':'吴慧芬','陈老':'陈岩石','老赵':'赵德汉','达康':'李达康','高总':'高小琴',

'猴子':'侯亮平','老郑':'郑西坡','小艾':'钟小艾','老师':'高育良','同伟':'祁同伟',

'赵公子':'赵瑞龙','郑乾':'郑胜利','孙书记':'孙连城','赵总':'赵瑞龙','昌明':'季昌明',

'沙书记':'沙瑞金','郑董':'郑胜利','宝宝':'张宝宝','小高':'高小凤','老高':'高育良',

'伯仲':'杜伯仲','老杜':'杜伯仲','老肖':'肖钢玉','刘总':'刘新建',"美女老总":"高小琴"}

names={} #姓名字典

relationships ={} #关系字典

lineNames =[] #每段内人物的关系

node=[] #存放处理后的人物

def read\_txt(path): #读取剧作并分词

jieba.load\_userdict("person.txt") #加载人物字典(注意这个文件要用utf-8编码，可以使用sublime进行转换为utf-8编码)

f=codecs.open(path,'r',encoding="utf-8") #读取剧作,并将其转换为utf-8编码

for line in f.readlines():

poss=pseg.cut(line) #分词并返回该词词形

lineNames.append([]) #为新读入的一段添加人物名称列表

for w in poss:

if w.word in stopwords: #去掉某些停用词

continue

if w.flag != "nr" or len(w.word) <2 :

if w.word not in replace\_words:

continue

if w.word in replace\_words: #将某些在文中人物的昵称替换成正式的名字

w.word=replace\_words[w.word]

lineNames[-1].append(w.word) #为当前段增加一个人物

if names.get(w.word) is None: #如果这个名字从来没出现过，初始化这个名字

names[w.word] =0

relationships[w.word] ={}

names[w.word] +=1 #该人物出现次数加1

for line in lineNames: #通过对于每一段段内关系的累加，得到在整篇小说中的关系

for name1 in line:

for name2 in line:

if name1 == name2:

continue

if relationships[name1].get(name2) is None: #如果没有出现过两者之间的关系，则新建项

relationships[name1][name2] =1

else:

relationships[name1][name2] +=1 #如果两个人已经出现过，则亲密度加1

def write\_csv():

# 在windows这种使用\r\n的系统里，不用newline=‘’的话

# 会自动在行尾多添加个\r，导致多出一个空行，即行尾为\r\r\n

csv\_edge\_file = open("edge.csv", "w", newline="")

writer = csv.writer(csv\_edge\_file)

writer.writerow(["source", "target", "weight","type"]) # 先写入列名,"type"为生成无向图做准备

for name,edges in relationships.items():

for v,w in edges.items():

if w>20:

node.append(name)

writer.writerow((name,v,str(w),"undirected")) # 按行写入数据

csv\_edge\_file.close()

#生成node文件

s=set(node)

csv\_node\_file =open("node.csv","w",newline="")

wnode =csv.writer(csv\_node\_file)

wnode.writerow(["ID","Label","Weight"])

for name,times in names.items():

if name in s:

wnode.writerow((name,name,str(times) ) )

csv\_node\_file.close()

if \_\_name\_\_=='\_\_main\_\_':

file = "剧情梗概.txt"

edge\_file="edge.txt"

read\_txt(file)

write\_csv()