政府新闻分析工具 gov_news_analysis 用户手册

崔冠宇 / Guanyu Cui
(cuiguanyu@ruc.edu.cn)

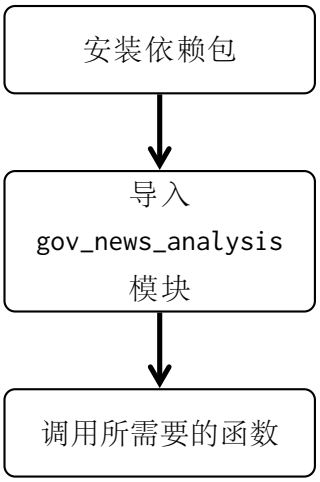
December, 2020

简介

gov_news_analysis 是由崔冠宇开发的一套政府新闻分析工具包，实现了文本分析与实体提取、社交网络构建以及多种网络指标的计算功能，具有良好的通用性与可用性。

目录

使用流程图



1 安装依赖包

本工具依赖下列 Python 包：

1. jieba[1]
2. pkuseg[2]
3. thulac[3]
4. lac[4]
5. networkx[5]
6. python-louvain[6]
7. pandas[7]

[1] https://github.com/fxsjy/jieba
[2] https://github.com/lancopku/pkuseg-python
[3] https://github.com/thunlp/THULAC-Python
[4] https://github.com/baidu/lac
[5] https://github.com/networkx/networkx
[6] https://github.com/taynaud/python-louvain
[7] https://github.com/pandas-dev/pandas

8. tabulate[8]

> 您可以在终端中使用下面的 pip 命令来安装各依赖包：
>
> ```
> $ pip install jieba pkuseg thulac lac
>   networkx python-louvain pandas tabulate
> ```
>
> 如果您需要更详细的说明或帮助，请参阅这些包的 github 页面。

2 导入 gov_news_analysis 模块

1. 将 gov_news_analysis.py 文件放在您需要调用本模块的 Python 文件所在的目录；

2. 在您的 Python 文件头部加入下列代码：

```
import gov_news_analysis as gna
```

如图1所示：

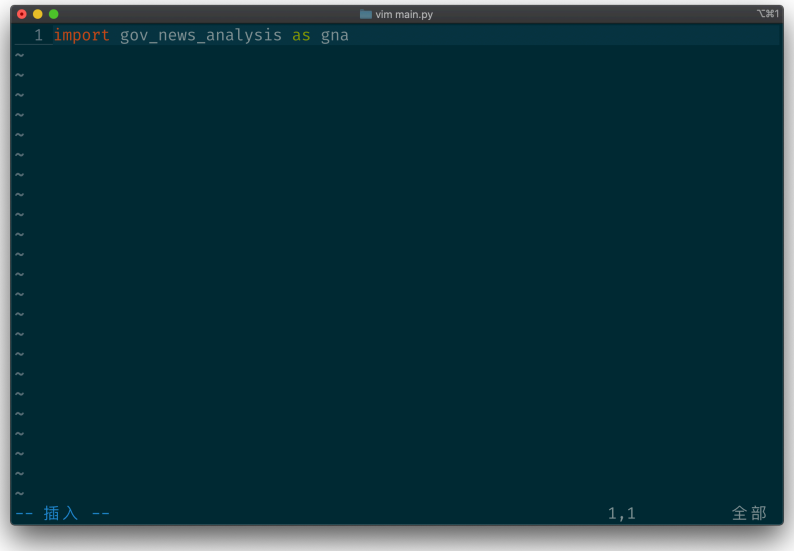


图 1: 在 Python 文件中导入模块

[8] https://github.com/astanin/python-tabulate

请注意：由于 gov_news_analysis 模块并非以 pip 方式安装，因此一定要将 gov_news_analysis.py 文件与您的代码放置在同一个目录下。或者您也可以自行将 gov_news_analysis.py 文件拷贝到您的 Python 包的安装目录下。

3 调用所需要的函数

3.1 数据预处理相关函数

在对社交网络进行统计分析前，需要先对新闻文本文件进行预处理，包括分词、实体识别与提取和社交网络构建等步骤。下面介绍有关数据预处理的函数：

3.1.1 gov_news_analysis.fetch_entities

- 原型：`fetch_entities(news_file, save_file = 'default_entity_file.txt', mode = 'jieba')`
- 功能：对按一定格式存储的新闻文本文件进行分词和实体识别，将实体及其词频以及每条新闻出现的人物保存至文本文件中，供进一步处理。
- 参数：
  - news_file(*string*) - 新闻文本文件，格式为每行一条记录，包括新闻链接（url）、发表时间（date）、来源（meta）、标题（title）以及正文内容（text）五个字段，各个字段用水平制表符（\t）隔开；
  - save_file(*string*, default: 'default_entity_file.txt') - 生成的实体文件的文件名；
  - mode(*string*, default: 'jieba') - 分词所用的中文处理包，可用的值有'jieba'、'pkuseg'、'thulac' 以及'baidu'。
- 返回值：
  - entity_freq(*dict*) - 实体 (*string*) 到词频 (*integer*) 的字典。
  - set_news_people(*dict*) - 新闻编号 (*integer*) 到人物集合 (*set*) 的字典。
- 备注：请尽量存储为 txt 格式的文件。

3.1.2 gov_news_analysis.most_frequently_entities_topk

- 原型：`most_frequently_entities_topk(entity_freq, top_k = 10, return_tabulate = True)`
- 功能：根据得到的实体-词频字典，返回词频前 k 高的实体。
- 参数：
  - `entity_freq`(*dict*) - 实体-词频字典；
  - `top_k`(*integer*, default: 10) - 求词频前 k 高的实体；
  - `return_tabulate`(*bool*, default: True) - 是否以表格形式返回。若是，返回 `tabulate` 对象；否则返回词频前 k 高的实体-词频字典。
- 返回值：
  - （若 `return_tabulate` 传入 `True`）`tabulate` - 词频前 k 高的实体-词频表格。
  - （若 `return_tabulate` 传入 `False`）`topk_entities_dict`(*dict*) - 词频前 k 高的实体-词频字典。
- 备注：无。

3.1.3 gov_news_analysis.build_social_network

- 原型：`build_social_network(set_news_people)`
- 功能：根据得到的新闻编号-人物集合字典，返回建立的社交网络图。
- 参数：
  - `set_news_people`(*dict*) - 新闻编号-人物集合字典；
- 返回值：
  - `G`(*NetworkX graph*) - 社交网络图。
- 备注：无。

3.1.4 gov_news_analysis.preprocess

- 原型：`preprocess(news_file, entity_file = 'default_entity_file.txt', network_file = 'default_network_file.txt', cut_mode = 'jieba')`
- 功能：对新闻文本文件执行上述三个函数，打印中间结果，保存建立的社交网络图为邻接表文件，并且返回社交网络图。
- 参数：
  - `news_file`(*string*) - 新闻文本文件，格式为每行一条记录，包括新闻链接（url）、发表时间（date）、来源（meta）、标题（title）以及正文内容（text）五个字段，各个字段用水平制表符（`\t`）隔开；
  - `entity_file`(*string*, default: 'default_entity_file.txt') - 实体文件的文件名（若文件存在，则会读取该文件，跳过分词与实体提取过程）；
  - `network_file`(*string*, default: 'default_network_file.txt') - 图的邻接表文件的文件名（若文件存在，则会读取该文件，跳过社交网络构建过程）；
  - `cut_mode`(*string*, default: 'jieba') - 分词所用的中文处理包，可用的值有'jieba'、'pkuseg'、'thulac' 以及'baidu'。
- 返回值：
  - `G`(*NetworkX graph*) - 社交网络图。
- 备注：若实体文件存在，则会利用该文件，跳过分词与实体提取过程；若图的邻接表文件存在，则会利用该文件，跳过社交网络构建过程。

3.2 图的基础分析相关函数

在构建完社交网络图后，还需要对图的基础指标进行计算和分析，主要包括与某节点联系最强的若干节点、图的基本统计数据以及 PageRank 影响力最高的若干节点等功能。下面介绍有关图的基础分析的函数：

3.2.1 gov_news_analysis.strongest_neighbors_topk

- 原型：`strongest_neighbors_topk(graph, node, top_k = 10, return_tabulate = True)`

- 功能：返回图上与某节点关系（按权重，即共现次数计）最强的 k 个节点。
- 参数：
  - graph(*NetworkX graph*) - 社交网络图；
  - node(*string*) - 节点名称；
  - top_k(*integer*, default: 10) - 求共现次数前 k 高的节点；
  - return_tabulate(*bool*, default: True) - 是否以表格形式返回。若是，返回 tabulate 对象；否则返回词频前 k 高的节点-共现次数字典。
- 返回值：
  - （若 return_tabulate 传入 True）tabulate - 共现次数前 k 高的节点-共现次数表格。
  - （若 return_tabulate 传入 False）topk_neighbor_dict(*dict*) - 共现次数前 k 高的节点-共现次数字典。
- 备注：若传入的参数中的节点在图上不存在，则输出提示信息并返回 None。

3.2.2 gov_news_analysis.graph_statistics

- 原型：graph_statistics(graph)
- 功能：打印并返回图的统计数据，包括节点数、边数、连通分量数以及最大连通分量节点个数。
- 参数：
  - graph(*NetworkX graph*) - 社交网络图；
- 返回值：
  - nodes(*integer*) - 图的节点数；
  - edges(*integer*) - 图的边数；
  - components(*integer*) - 图的连通分量数；
  - largest_comp(*integer*) - 图的最大连通分量的节点数；
- 备注：无。

3.2.3 gov_news_analysis.pagerank_influence_topk

- 原型：`pagerank_influence_topk(graph, top_k = 20, return_tabulate = True)`
- 功能：返回图上 PageRank 值最高的 k 个节点。
- 参数：
  - `graph`(*NetworkX graph*) - 社交网络图；
  - `top_k`(*integer*, default: 20) - 求 PageRank 前 k 高的节点；
  - `return_tabulate`(*bool*, default: True) - 是否以表格形式返回。若是，返回 `tabulate` 对象；否则返回 PageRank 前 k 高的节点-PageRank 字典。
- 返回值：
  - （若 `return_tabulate` 传入 `True`）`tabulate` - PageRank 值前 k 高的节点-PageRank 表格。
  - （若 `return_tabulate` 传入 `False`）`topk_influence_dict`(*dict*) - PageRank 值前 k 高的节点-PageRank 字典。
- 备注：无。

3.2.4 gov_news_analysis.basic_analysis

- 原型：`basic_analysis(graph)`
- 功能：对构建出的社交网络图执行上述三个函数，并且打印中间结果。
- 参数：
  - `graph`(*NetworkX graph*) - 社交网络图；
- 返回值：无。
- 备注：无。

3.3 图的进阶分析相关函数

在上面的图的基础分析部分，仅仅统计分析了图的简单指标，下面还可以对图的结构进行深入研究，主要包括小世界理论验证、三元闭包验证、社区挖掘、中介中心性计算、聚集系数计算以及结构洞挖掘等步骤。下面介绍有关图的进阶分析的函数：

3.3.1 gov_news_analysis.smallworld_validate(未实现)

- 原型：`smallworld_validate(graph)`
- 功能：验证图是否满足小世界现象。
- 参数：
  - graph(*NetworkX graph*) - 社交网络图；
- 返回值：无。
- 备注：无。

3.3.2 gov_news_analysis.ternary_closure_validate(未实现)

- 原型：`ternary_closure_validate(graph)`
- 功能：验证图是否满足三元闭包性质。
- 参数：
  - graph(*NetworkX graph*) - 社交网络图；
- 返回值：无。
- 备注：无。

3.3.3 gov_news_analysis.community_detection

- 原型：`community_detection(graph, top_k = 5, print_all = False, mode = 'louvain')`
- 功能：打印并返回图中最大的 k 个社区。
- 参数：

  - graph(*NetworkX graph*) - 社交网络图；
  - top_k(*integer*, default: 5) - 求最大的 k 个社区；
  - print_all(*bool*, default: False) - 是否打印社区内的所有节点。若是，对每个社区输出所有节点；否则对每个社区输出一个代表元。
  - mode(*string*, default: 'louvain') - 社区检测算法，可用的值有'louvain'、'asyn_lpa' 以及'lpa'。
- 返回值：
  - （若 mode == 'louvain'）partition(*dict*) - 节点-所属社区字典。
  - （若 mode 为其它两种）partition_gen(*generator*) - 社区节点生成器。
- 备注：在 print_all == False 时，每个社区的代表元指 PageRank 值最大的节点。

3.3.4 gov_news_analysis.centrality_topk

- 原型：centrality_topk(graph, sample_ratio = 0.05, top_k = 10, return_tabulate = True)
- 功能：返回图上中介中心性最高的 k 个节点。
- 参数：
  - graph(*NetworkX graph*) - 社交网络图；
  - sample_ratio(*float*, default: 0.05) - 近似求解时的节点采样率；
  - top_k(*integer*, default: 10) - 求中介中心性前 k 高的节点；
  - return_tabulate(*bool*, default: True) - 是否以表格形式返回。若是，返回 tabulate 对象；否则返回中介中心性前 k 高的节点-中介中心性字典。
- 返回值：
  - （若 return_tabulate 传入 True）tabulate - 中介中心性前 k 高的节点-中介中心性表格。
  - （若 return_tabulate 传入 False）topk_centrality_dict(*dict*) - 中介中心性前 k 高的节点-中介中心性字典。
- 备注：无。

3.3.5 gov_news_analysis.clustering_coefficient_topk

- 原型：`clustering_coefficient_topk(graph, top_k = 10, return_tabulate = True)`
- 功能：返回图上聚集系数最高的 k 个节点。
- 参数：
  - `graph`(*NetworkX graph*) - 社交网络图；
  - `top_k`(*integer*, default: 10) - 求聚集系数前 k 高的节点；
  - `return_tabulate`(*bool*, default: True) - 是否以表格形式返回。若是，返回 `tabulate` 对象；否则返回聚集系数前 k 高的节点-聚集系数字典。
- 返回值：
  - （若 `return_tabulate` 传入 `True`）`tabulate` - 聚集系数前 k 高的节点-聚集系数表格。
  - （若 `return_tabulate` 传入 `False`）`topk_cc_dict`(*dict*) - 聚集系数前 k 高的节点-聚集系数字典。
- 备注：无。

3.3.6 gov_news_analysis.structural_holes_detection(未实现)

- 原型：`structural_holes_detection(graph)`
- 功能：挖掘图上的结构洞。
- 参数：
  - `graph`(*NetworkX graph*) - 社交网络图；
- 返回值：无。
- 备注：无。

3.3.7 gov_news_analysis.optional_analysis

- 原型：optional_analysis(graph)
- 功能：对构建出的社交网络图执行上述六个函数，并且打印中间结果。
- 参数：
  - graph(*NetworkX graph*) - 社交网络图；
- 返回值：无。
- 备注：在执行社区检测算法时，测试三种不同算法的效果。

4 实际案例演示

下面用政府新闻社交网络分析的示例演示如何调用本模块。新建一个 Python 文件，键入下列代码（路径请替换为您的文件放置位置）：

```
1  import networkx as nx
2  import gov_news_analysis as gna
3
4  print('一、数据预处理:')
5  G = gna.preprocess(
6      news_file = '../../data/gov_news.txt',
7      entity_file = '../../data/
       entities_data_baidu.txt',
8      network_file = '../../data/
       network_data_baidu.txt',
9      cut_mode = 'baidu')
10 print('保存gexf文件中...')
11 nx.write_gexf(G, '../../data/graph.gexf')
12 print('保存完成')
13 print('')
14 print('二、基础分析:')
15 gna.basic_analysis(G)
16 print('')
17 print('三、可选分析:')
```

```
18 gna.optional_analysis(G)
```

然后执行 Python 程序，发现可以正常运行，能输出结果。如图 2 所示。

图 2: 政府新闻社交网络分析案例运行测试