Python程序设计与数据科学导论课程期中大作业-文本检索实验报告

1900017823 文天宇

一.基础要求实现情况

1. 数据处理（20'）：
    - `process_news.py`文件内实现了新闻数据的预处理（包含大小写转换、停用词过滤、分词、低频词过滤和词形统一化等操作）并构建词汇表。运行程序后保存在当前目录下的`processed_news.csv`文件包含了预处理后的新闻数据，保存在当前目录下的`vocab.txt`文件作为词汇表以备后续文本检索使用，保存在当前目录下的`words_in_news.csv`文件包含了新闻的分词结果（已去除低频词，故保证词语均在词汇表中）以备后续TF-IDF构造使用。
2. 检索排序（30'）：
    - 为实现C/S架构，`local_server.py`文件内实现了服务端功能（包含连接客户端、接收客户端发送的检索请求、多线程并发处理检索请求和发送给客户端检索结果）。
    - `GUI.py`文件内实现了客户端及图形交互界面功能（包含连接服务端、发送给服务端检索请求、接收服务端发送的检索结果和通过图形交互界面引导用户检索并展示检索结果）。
    - 检索结果的初步排序策略可以为按照包含对应词语数目排序等，由于其后续被HITS算法替代，故最终版本不再呈现该部分的实现。
3. 排序优化（20'）：
    - `get_tfidf.py`文件内实现了新闻中词语的TF_IDF值的计算，具体计算方法见代码实现。运行程序后保存在当前目录下的`tf_idf.csv`文件包含了所有新闻中所有词语的TF_IDF值结果。
    - `get_news_word_matrix.py`文件内实现了新闻-词汇TF-IDF矩阵的构建和降维处理，降维算法采用PCA降维，保留主成分数目默认为100。运行程序后保存在当前目录下的`tfidf_matrix.npy`文件记录了原始的新闻-词汇TF-IDF矩阵，而`PCAed_tfidf_matrix.npy`文件记录了降维后的矩阵。
    - `search_and_sort_utils.py`文件内定义和实现了用于新闻检索的检索函数和检索结果排序函数，所有函数的参数列表及返回值具体定义见代码注释。其中，`find_news_list_by_word()`和`find_news_list_by_words_list()`分别用于检索单个词语所在新闻和多个词语同时出现的新闻，`hitsrank_by_news_list()`借助上面得到的TF_IDF矩阵计算所有新闻向量的余弦相似度，进而构建以新闻为结点、以相似度为边权的无向有权重图，在该图上执行HITS算法对检索结果进行排序，`search_and_sort_by_word()`对上述三个子函数进行包装，自动识别传进参数为词语或词语列表，先执行检索操作，再返回排序后的检索结果。
    - 此处检索结果的优化还可以使用PageRank算法，但在实际操作中发现，由于新闻数目过少，TF_IDF矩阵稀疏程度太高，即使进行降维处理后执行PageRank算法依旧难以收敛，故此处不再使用该算法。
4. 文章聚类（10'）：

- `news_cluster.py`文件内实现了新闻聚类以验证之前得到的新闻TF-IDF向量的合理性。其中`map_labels()`用于实现聚类结果和真实标签的最优匹配，`purity()`用于计算聚类结果的纯度，其参数列表及返回值具体定义见代码注释。首先根据降维处理后的TF-IDF向量执行K-means聚类，目标类别数目对应真实标签数目（5），然后对聚类结果执行最优匹配，最后计算纯度值。运行程序会打印纯度值计算结果（最高可以达到0.95）。

5. 相似词（10'）：
   - `find_synonym.py`文件内实现了相似词的查找，查找策略为首先通过对前面得到的原始TF-IDF矩阵做转置得到每行作为词语向量，之后对词语向量做PCA降维处理（保留主成分数目默认仍为100），然后计算词语向量的余弦相似度，对每个词语下和其他词语的相似度做排序，取出前k个词语视为其相似词（k默认为2）。运行程序后保存在当前目录下的`synonym_indices_k.npy`文件记录了每个词语相似词对应的下标，而`synonym.txt`文件直接呈现出了每个词语及其对应的相似词识别结果。在该相似词查找策略下，某个词语的相似词更倾向于和该词语出现在同样的新闻中。
   - `search_and_sort_utils.py`文件内的函数`get_synonyms()`用于在服务端通过索引下标方便地查找给定词语的相似词（可以指定k值）。
6. 模糊匹配（10'）：
   - 见下文模糊检索部分。

二.加分项实现情况

1. 在图形交互界面增加检索方法选项（10'）：

   当前可以使用的检索方法有：

   - 宽松检索：输入1-3个检索词，输出的新闻中包含其中任意词语即可。更具体地，算法优先检索并排序包含检索词数目较多的新闻，因而只包含其中单个词语的新闻将被置于检索结果的较后部分。
   - 严格检索：输入1-3个检索词，输出的新闻中同时包含其中所有词语。
   - 模糊检索：输入1个检索词，输出的新闻中包含该词语或该词语的相似词。更具体地，算法优先检索并排序包含该检索词的新闻，因而包含相似词的新闻将被置于检索结果的较后部分

   在代码中，上述三种检索方法通过在客户端向服务端发送的检索请求中加入不同的模式标记实现。值得注意的是，上述不同部分的检索结果可能出现重复，这里采取只保留每条检索结果最前方的一条的策略（实现在`local_server.py`文件中的线程内检索部分）。三种检索在图形交互界面的最终显示效果如下：

2. 检索结果不同排序策略的比较（5'）：

   备选的排序策略有：

   - 朴素算法：检索词出现次数越多，排序位置越靠前。
   - HITS算法：在有权重图上利用不同新闻的内容权威度Authority和链接权威度Hub来对新闻质量进行评估，选取前者进行排序，即内容权威度Authority越高，排序位置越靠前。
   - PageRank算法：基于随机游走的马尔科夫链转移矩阵，通过迭代直至收敛的方法得到新闻质量的排序结果。

   通过实验可以看到，朴素算法虽然较为简单，但只通过检索词出现次数进行排序的策略过于天真，而PageRank算法虽然对新闻质量的评估结果较为优秀，但收敛速度过慢，尤其对于数据集中新闻数量过少的情况更甚。综合比较而言，HITS算法计算复杂度适中，且收敛条件易于达成，故最终选取该算法进行检索结果的排序。

三. 部分运行结果展示

1. 词汇表部分保存结果：

```
8433    played
8434    player
8435    playground
8436    playing
8437    playlist
8438    playstation
8439    playwright
8440    plea
8441    plead
```

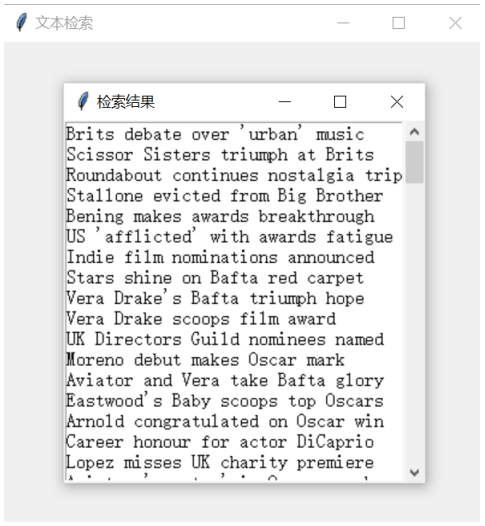
2. TF-IDF值计算部分结果：

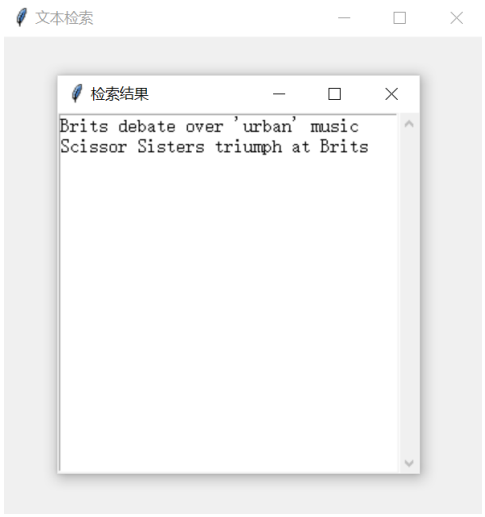
id	level_1	word	TF	IDF	TF-IDF
1	0	quarterly	0.004201680672268907	4.711779921046349	0.019797394626245163
1	1	profit	0.037815126050420166	2.9117216490035998	0.11010712118080838
1	2	u	0.012605042016806723	0.9552418184585985	0.012040863257881493
1	3	medium	0.004201680672268907	2.3604046638828717	0.009917666654970049
1	4	giant	0.004201680672268907	2.638607992380109	0.01108658820327777
1	5	timewarner	0.029411764705882353	7.707512194600341	0.22669153513530413
1	6	jumped	0.004201680672268907	4.273524990115194	0.017955987353425185
1	7	bn	0.02100840336134454	1.9114544438349685	0.04015660596291951
1	8	three	0.008403361344537815	1.3055949978731551	0.010971386536749202
1	9	month	0.004201680672268907	1.3105825393841943	0.005506649325143673
1	10	december	0.004201680672268907	2.3230171318112514	0.009760576184080888
1	11	year	0.01680672268907563	0.4254385365068759	0.007150227504317242
1	12	earlier	0.008403361344537815	2.135358162422576	0.017944186238845174
1	13	firm	0.004201680672268907	1.5784619845397954	0.006632193212352081
1	14	one	0.008403361344537815	0.7135792193771513	0.005996464028379422
1	15	biggest	0.004201680672268907	2.190059298135633	0.009201929824099297
1	16	investor	0.004201680672268907	3.092391677759081	0.01299324234352555
1	17	google	0.008403361344537815	4.096594281956116	0.034425162033244666

3. 新闻TF-IDF向量余弦相似度矩阵部分结果可视化：

	0	1	2	3	4	5	6	7	8	9
0	1.00000	0.03730	0.19944	0.70513	0.24563	-0.08570	-0.08893	-0.01189	0.43480	0.01649
1	0.03730	1.00000	0.02450	0.03880	0.13818	-0.07062	-0.06096	0.01846	0.05820	-0.03581
2	0.19944	0.02450	1.00000	0.16078	0.12692	0.09194	-0.05392	-0.02374	0.10394	0.08512
3	0.70513	0.03880	0.16078	1.00000	0.27694	-0.01456	-0.10904	0.02690	0.39625	0.00183
4	0.24563	0.13818	0.12692	0.27694	1.00000	-0.10287	-0.23553	0.19060	0.32307	0.00311
5	-0.08570	-0.07062	0.09194	-0.01456	-0.10287	1.00000	0.62711	-0.07125	-0.03730	-0.04301
6	-0.08893	-0.06096	-0.05392	-0.10904	-0.23553	0.62711	1.00000	-0.00175	-0.05809	-0.03990
7	-0.01189	0.01846	-0.02374	0.02690	0.19060	-0.07125	-0.00175	1.00000	0.01613	0.03370
8	0.43480	0.05820	0.10394	0.39625	0.32307	-0.03730	-0.05809	0.01613	1.00000	0.07985
9	0.01649	-0.03581	0.08512	0.00183	0.00311	-0.04301	-0.03990	0.03370	0.07985	1.00000

4. 检索过程图形交互界面部分显示结果（从上至下依次为检索词、宽松检索结果和严格检索结果）：





5. 相似词表部分保存结果：

```
rated: sublime restrained
rather: overly prerogative
ratification: treaty constitution
ratified: blackpool bournemouth
rating: pg skywalker
rational: instinct pandering
```

*为了方便起见，上述过程中保存的较大文件和原始数据集文件均未随代码和报告一同上交。如果需要在没有这些数据的情况下重新运行并生成数据，请最好按照上面提及的步骤顺序运行程序，否则可能出现文件无法找到的错误。