COL-764 Assignment 3
PageRank for Document Collections.

Submitted by-Dipanshu Patidar (2018CS50405)

File Details

- simgraph_gen.py :- only file that implements the required task, i.e. takes input, calculate similarity and then compute pageranks.

Implementation Details

We needed to implement following subtasks in this assignment, implementation of this subtasks are breifly explained below:

for this, I take 3 inputs as mentioned in the assignment description. First is the nname of the method used for similarity computation, second is the path to th collection folder, and third is the name of the output file.
After taking the input, I make two lists namely file_paths and file_names. After this I traverse through the collection folder and extract the paths and names all files, and then append this paths and names into respective lists.

Then I call process function to compute similarity between the documents and generate a list of tuples (i,j,sim_i_j), for each pair of documents.
In this process function, first I check what type of method is to be used for similarity calculation, then preprocessing of the documents is done, after this for each type of method is handled differently.

For jaccard similarity, set of tokens of preprocessed documents are formed and to calculate jaccard similarity between 2 documents, simply sizeof(intersection)/sizeof(union) is returned.

For cosine similarity, we make tf_idf vectors of all files and the queries and then ccompute the cosine similarity between them. for tf_idf vectorization, we have a dictionary called global_dict , this dictionary contain all the word present in all files(i.e. vocabulary) , here key is a word and corresponding value is number of documents in dataset that contain that word . This will give us the idf of each word. For term frequency, we make a dictionary corresponding to each file, each of this dictionary is (key:word,value:number of times this word appears in this document), so we get term frequency for each word in each file. Then We just multiply tf and idf to get tf_idf vector for each file. o We do same for each query to get tf_idf vector for each query. o We then use helper function getCosine to get the cosine similarity between any two vectors(here we have dictionaries).

In both cases after calculating similarity, the similarity along with corresponding document indices was appended in the list that is to be returned.

Here first part is done.

For the second part, first I make graph with file indices as nodes and their similarities as edge weights. I use **edgelist2adjacency** function of sknetwork library for making the graph.
After this pagerank() and fit_transform(graph) functions of sknetwork are used to compute page ranks.

At last sorting is done and we obatin documents(pages) in descending order of their pageranks.

1)What was the approach taken to adapt the PageRank algorithm to a weighted, undirected graph? Also, describe your method of choice for computing PageRank scores and your experience.

We studied PageRank as being computed on a directed, unweighted web graph. Here, however, we have an undirected, weighted graph between documents where the weights are based on the corresponding Sim score computed between two documents.So we need to take care of two things in our new algorithm to compute pagerank

1. undirected graph instead of directed graph
2. edges do not have same weight

To deal with the first problem, we know that any undirected graph can be converted into a directed graph with two directed edges for each undirected edge between nodes. So 1st change is easily accomodated here. For the second one, the transition, and the random teleportation probabilities are both multiplied by the edge weight and normalized again to sum up to 1. This way, PageRank can be used for the given application.

I experimentied with two approaches
i) networkx library ii) sknetwork library to calculate the pagerank.
Sknetwork library is faster. Hence, I implemented this in the submission.

PageRank computes ranking of the nodes (documents) in the graph G based on the structure of the incoming links.
It is based on using a transition matrix and current state probability vector.
Let probability vector is X = (x1, x2, ..., xn)
In the transition probability matrix P, the ith row tells us where we go next from state i.
The detailed description of this algorithm can be found here:
https://www.geeksforgeeks.org/pagerank-algorithm-implementation/

2) List of documents and their PageRank scores with top-20 highest PageRank values for each of the similarity functions.

- **Top 20 documents for cosine similarity with their page rank scores:**
  - talk.politics.misc/179058 0.000357
  - talk.politics.misc/179073 0.000349
  - talk.politics.misc/178908 0.000338
  - soc.religion.christian/21496 0.000326
  - comp.graphics/39638 0.000325
  - comp.graphics/39078 0.000325
  - comp.sys.mac.hardware/52004 0.000319
  - talk.politics.misc/179034 0.000314
  - talk.politics.mideast/77195 0.000313
  - talk.politics.misc/179054 0.000311
  - talk.politics.mideast/76479 0.000309
  - talk.politics.mideast/77198 0.000303
  - comp.windows.x/67961 0.000301

- comp.os.ms-windows.misc/10034 0.000295
- talk.religion.misc/84223 0.000292
- talk.politics.mideast/77186 0.000292
- soc.religion.christian/21597 0.000288
- soc.religion.christian/21748 0.000288
- talk.politics.mideast/77397 0.000288
- soc.religion.christian/21744 0.000287

- **Top 20 documents for jaccard similarity with their page rank scores:**
  - sci.electronics/54247 0.000177
  - sci.med/59271 0.000171
  - sci.med/59454 0.000171
  - talk.religion.misc/84349 0.000169
  - rec.autos/103727 0.000169
  - sci.med/59407 0.000167
  - alt.atheism/54160 0.000167
  - rec.sport.baseball/104999 0.000167
  - comp.sys.ibm.pc.hardware/60807 0.000166
  - rec.sport.baseball/104986 0.000166
  - comp.os.ms-windows.misc/10201 0.000166
  - sci.electronics/54164 0.000166
  - comp.sys.ibm.pc.hardware/60804 0.000166
  - talk.politics.guns/54554 0.000166
  - comp.sys.mac.hardware/52241 0.000165
  - comp.sys.mac.hardware/52443 0.000165
  - soc.religion.christian/21586 0.000165
  - rec.sport.hockey/54735 0.000165
  - comp.graphics/38863 0.000164
  - sci.electronics/54208 0.000164

3)What specific conclusions you can draw on the kind of documents that have high PageRank scores?

1. In case of top20 documents obtained from TF-IDF based cosine similarity, we can clearly see two clusters of documents in the top 20 documents. One of them corresponds to politics and other corresponds to religion.
2. Fourth document soc.religion.christian/21496 is a FAQ essay on homosexuality. There is a possibility that many people discussed this document and was cited in many news articles.
3. Many top documents are from talk.politics.nisc and THE WHITE HOUSE office of the Press Secretary. There is a possibility that other documents referred to these documents and talked about them a lot.

Some of the conclusions drawn are the following-

• Trusted sources of information such as government sources which are a matter of public debate rank highly.

• Email exchanges on a particular topic with essay-type standard replies which is cited by many others in a group has also achieved high page rank scores.
• The 7 th ranked document related to sys.mac.hardware (only document from such sub-collection) has high pagerank score since it is FAQ (Frequently Asked Questions) page which clarifies doubts on Macintoshes. It is more likely many documents on hardware may cite this information. Similarly, the 12th ranked document related to comp.windows.x is also a FAQ page. In cases of top20 documents for Jaccard similarity, no such insights as above could be found. The documents seemed to be just mails/ replies / short documents on various topics. No clusters of documents were found. Even documents from same sub-collection didn't have same topic. As we can see, top 20 documents retrieved from cosine similarity based pagerank provides more meaningful results in terms of explaining the reason behind high ranks of the obtained pages and their relation whereas not many conclusions can be drawn from the top 20 pages of jaccard based pagerank.

My conclusion

1)Jaccard gave short randon documents. It was och harder to draw any conclusion. Also, you can see that all of these 2 documents have almost the same page rank scores which do not indicate clear ranks.
2)In my opinion cosine works much better for page rank.
3) The documents(which are news articles mainly) that are cited(or dicussed) by large number of agencies and people are the ones that are on top.
4) Many of the documents in top 20 of cosine are large documents, this can be attributed to large probabilty of finding same word in other document and hece more similarity.