AI607: Graph Mining and Social Network Analysis (Fall 2024)

Homework 2: Citation Network Analysis using PageRank and HITS

Release: October 4, 2024,
Due: October 18, 2024, 11:59pm

In this assignment, your tasks are to implement PageRank and HITS algorithms for measuring the importance of web pages and to analyze a citation network using these algorithms. More information about PageRank and HITS can be found in the lecture note and Chapter 5 of MMDS. (`http://infolab.stanford.edu/~ullman/mmds/ch5.pdf`).

1 Introduction

The PageRank score reflects the stationary probability of a web surfer being located on each page. The web surfer navigates between web pages according to the following rule: the surfer begins at a random web page and, at each page, has the option to either jump to a random page (with probability $1 - \beta$) or follow a link from the current page (with probability β). We refer to β as the damping factor. When the current web page lacks any links, the surfer always jumps to a random page.

While PageRank considers page importance as a single-dimensional concept, HITS recognizes two distinct types of importance for pages: Hub score and Authority score. Some web pages, called hubs, are important not because they provide information but because they link to informative web pages called authorities. In other words, a good hub is a page with links to many good authorities, and a good authority is a page linked to by many hubs. Thus, web pages can be evaluated based on two scores, one for their quality as a hub and the other for their quality as an authority. For each node, its Hub score is obtained by summing the Authority scores of its neighbors. Next, the Authority score of each node is obtained by summing all of its neighbors' Hub scores. Repeating these two steps with normalization eventually leads to the final scores.

2 Definition

2.1 PageRank

Let the PageRank score of node i at epoch t be $r_i^{(t)}$. Then, the PageRank score of node i at epoch $t+1$ is computed as follows:

$$r_i^{(t+1)} = \beta \sum_{j \to i} \frac{r_j^{(t)}}{d_j} + (1-\beta)\frac{1}{N} + \frac{\beta}{N} \sum_{j \in \{x | d_x = 0\}} r_j^{(t)}, \tag{1}$$

where β is a damping factor, N is the total number of nodes, and d_j is the out-degrees of node j. In equation (1), the red term represents the contribution from nodes that are connected to node i including that of a random jump, while the blue term signifies the contribution from nodes without outgoing links. In this assignment, you should implement the PageRank algorithm, which repeats this process until $r^{(t)}$ converges. Then, you should apply the PageRank algorithm to a citation network, and report the top-10 PageRank scores with the corresponding node IDs. More details are provided in the next section.

2.2 HITS

Let the Authority score and the Hub score of node i at epoch t as $a_i^{(t)}$ and $h_i^{(t)}$. Then, the Hub and Authority score of node i at epoch $t+1$ are computed as follows:

$$h_i^{(t+1)} = \sum_{i \to j} a_j^{(t)}, \quad a_i^{(t+1)} = \sum_{j \to i} h_j^{(t+1)} \tag{2}$$

Then, $a^{(t+1)}$ and $h^{(t+1)}$ are normalized so that their L2 norms become 1. In this assignment, you should implement the HITS algorithm, which repeats this process until $a^{(t)}$ and $h^{(t)}$ converge. Then, you should apply the HITS algorithm to a citation network, and report the top-10 Hub scores and Authority scores with the corresponding node IDs. More details are provided in the following section.

3 Implementation

3.1 Preprocessing

Your first task is to complete the `CitationNetwork` class in `graph.py`. To do this, you have to parse the file which contains the citation data. We will provide you `graph.txt`[1], which contains $\sim 1,570,000$ blocks, each of which is for a paper. Each block is in the following format:

```
#* --- Paper title
#@ --- Authors
#year --- Year
#conf --- Publication venue
#citation --- Citation number (both -1 and 0 means none)
#index --- Index id of this paper
#arnetid ---- Pid in arnetminer database
```

[1]The provided dataset can be obtained from the DBLP citation network (version 5) dataset (`https://www.aminer.org/citation`).

```
#% --- The id of references of this paper
        (there are multiple lines, with each indicating a reference)
#! --- Abstract
```

When converting the file into a directed graph, consider each paper as a node, and each (paper, reference) pair as a directed edge.

3.2 PageRank

Fill out the `pagerank` function in `graph.py`, which is for computing the PageRank score using the power iteration method. In this homework, the PageRank score should be initialized uniformly. In addition, the iteration should stop when one of the following conditions is satisfied:

- The number of iterations exceeds the `max_iter`
- The L2-norm of the difference between the previous PageRank score and the new PageRank score is lower than the tolerance, i.e., $\left\|r^{(t+1)} - r^{(t)}\right\|_2 < tol$.

3.3 HITS

Next, fill out the `hits` function in `graph.py`, which is for computing the Hub and Authority scores using the power iteration method. In this homework, the Hub and Authority scores should be initialized uniformly. In addition, the iteration should stop when one of the following conditions is satisfied:

- The number of iterations exceeds the `max_iter`
- The L2-norm of the difference between previous Hub scores and new Hub scores is lower than the tolerance, i.e., $\left\|h^{(t+1)} - h^{(t)}\right\|_2 < tol$.

3.4 Top-k Selection

Finally, implement the `print_top_k` function in `graph.py`, which is for printing the top-k scores and their corresponding paper titles and IDs in descending order in the following format:

```
<PAPER ID 1><TAB><SCORE 1><TAB><PAPER TITLE 1>
<PAPER ID 2><TAB><SCORE 2><TAB><PAPER TITLE 2>
...
<PAPER ID K><TAB><SCORE K><TAB><PAPER TITLE K>
```

More information can be found in the skeleton codes. Your implementation should be compatible with Python 3.7 or 3.8. Furthermore, you are allowed to use `numpy` and `scipy`, but other external libraries are not allowed. Feel free to use any Python's default library.

4 Test

You can test your implementation by executing the `main.py` with some options:

```
usage: main.py [-h] [-f FILE]

Get the top 10 PageRank scores, Hub scores, and Authority scores
with corresponding nodes for the given graph

optional arguments:
  -h, --help            show this help message and exit
  -f FILE, --file FILE  A file path for an initial matrix
```

- If you add the argument '-f [file]', then the program will load the input file as a graph, and run HITS algorithm.
- Hyperparameter β: This represents the damping factor for PageRank, and is set to values $\{0.8, 0.85, 0.9\}$.
- Hyperparameter `TOL`: This denotes the tolerance for both PageRank and HITS algorithms, and is set to $1e^{-5}$.
- Hyperparameter `MAX_ITER`: This defines the maximum number of iterations for both PageRank and HITS algorithms, and is set to 100.
- Hyperparameter `K`: This specifies the number of top-k results to return, and is set to 10.

5 Analysis

- Using `main.py`, report the list of **paper titles** and **IDs** with the top-10 PageRank for each damping factor, the top-10 hub scores, and the top-10 authority scores in `graph.txt`.
- For the analysis of the PageRank algorithm, report the results with varying damping factors, $\beta \in \{0.8, 0.85, 0.9\}$.

6 Notes

- Your implementation should run on TA's desktop within 10 minutes.
- Do not change the default values of hyperparaemters: β, `TOL`, `MAX_ITER`, and `K`.
- You may encounter some subtleties when it comes to implementation. Come up with your own design and/or contact Heechan Moon (heechan9801@kaist.ac.kr) and Seokbum Yoon (jing9044@kaist.ac.kr) for discussion. Any ideas can be taken into consideration when grading if they are written in the *readme* file.
- Different implementations may give slightly different PageRank and Hub-Authority scores due to the randomness, floating point errors, and so on. In that reason, your implementation will be evaluated in a robust way. Specifically, the scores that your implementation gives and the ground-truth scores will be compared after we round both scores to the nearest ten thousandth.

7 How to submit your assignment

1. Create hw2-[your student id].tar.gz, which should contain the following files:
   - **main.py, graph.py**: these should contain your implementation.
   - **report.pdf**: this should contain your answers in Section 5 (Analysis).
   - **readme.txt**: this file should contain the names of any individuals from whom you received help, and the nature of the help that you received. That includes help from friends, classmates, lab TAs, course staff members, etc. In this file, you are also welcome to write any comments that can help us grade your assignment better, your evaluation of this assignment, and your ideas.
2. Make sure that no other files are included in the tar.gz file.
3. Submit the tar.gz file at KLMS (`http://klms.kaist.ac.kr`).