PageRank 作业报告

1400012923 关玉烁

一、实验环境及数据集

- 服务器:Linux netlabml-ThinkStation-D30 4.4.0-96-generic #119-Ubuntu SMP Tue Sep 12 14:59:54 UTC 2017 x86*64 x86*64 x86_64 GNU/Linux
- 编程语言：python
- 数据集：ANN（http://tangra.cs.yale.edu/newaan）

二、提交文件结构

- code：该文件夹包含本次作业全部代码。
  - authorRank.py：对author的pagerank算法文件。
  - paperRank.py：对paper的pagerank算法文件。
  - venueRank.py：对venue的pagerank算法文件。
- results：该文件夹包含排序后的paper/author/venue的PageRank值文件。
  - retAuthor.csv：author的姓名及对应pagerank值，按照pagerank值降序排列。
  - retPaper.csv：paper的标题及对应pagerank值，按照pagerank值降序排列。
  - retVenue.csv：venue的名称及对应pagerank值，按照pagerank值降序排列。
- 2014：该文件夹包含ANN数据集。
- 作业报告_1400012923_关玉烁.pdf：实验报告。

三、基本思路

1. 对author的pagerank：利用了`2014/networks/author-citation-network-nonself.txt`，该文件每一行形如`author1 ==> author2`,记录了作者之间的引用关系，且不包括自己对自己的引用。在具体操作中，首先使用函数split提取每一行的author1和author2，建立起作者间的引用关系图。之后，编写PageRank算法，算出作者的PageRank值。
2. 对paper的pagerank：利用了`2014/networks/paper-citation-network-nonself.txt`和`2014/acl-metadata.txt`，前者每一行形如`paper1 ==> paper2`，记录了paper之间的引用关系，且不包括自己对自己的引用；后者记录了每篇文章的id、author、title、venue和year信息。在具体操作中，首先使用函数split提取`2014/networks/paper-citation-network-nonself.txt`每一行的paper1和paper2，建立起作者间的引用关系图。之后，编写PageRank算法，算出作者的

PageRank值。由于`2014/networks/paper-citation-network-nonself.txt`中均为paper的id，还需利用`acl-metadata.txt`，找到paper id对应的标题。

3. 对venue的pagerank：利用了`2014/networks/paper-citation-network-nonself.txt`和`2014/acl-metadata.txt`。由于数据集中没有直接记录venue相互引用关系的文件，我们需要自行提取。我的方法是利用paper之间的引用来构造venue之间的引用。举例来说，假如EMNLP的100篇文章引用了ACL，那么就构造一条从节点V_{EMNLP} 到V_{ACL}权重为100的边。

四、实验结果分析

PageRank算法中，只有随机游走的概率（$1-\alpha$）的值是可以改变的，不同的应用实例可能选用不同的α，我在实验中分析了$\alpha=0.1,0.5,0.9$时算法运行的效果。

- author：三列分别为$\alpha=0.1,0.5,0.9$时，算法得到top3的作者姓名，以及我通过https://www.semanticscholar.org/查到的highly influential citations数量，我通过highly influential citations来判断pagerank得到的排名是否合理。从图中看到，$\alpha=0.1$时top 3的作者highly influential citations更多，pagerank效果更好。

$\alpha=0.1$	$\alpha=0.5$	$\alpha=0.9$
Manning, Christopher D.， 3102	Della Pietra, Vincent J.，583	Mercer, Robert L.，555
Church, Kenneth Ward，355	Church, Kenneth Ward，355	Della Pietra, Vincent J.，583
Klein, Dan，1014	Mercer, Robert L.，555	Brown, Peter F.497

- paper：三列分别为$\alpha=0.1,0.5,0.9$时，算法得到top3的paper标题，以及我通过https://www.semanticscholar.org/查到的highly influential citations数量，我通过highly influential citations来判断pagerank得到的排名是否合理。从图中看到，$\alpha=0.1$时top 3的paper highly influential citations更多，pagerank效果更好。

$\alpha=0.1$	$\alpha=0.5$	$\alpha=0.9$
Building A Large Annotated Corpus Of English: The Penn Treebank，474	Building A Large Annotated Corpus Of English: The Penn Treebank，474	A Stochastic Parts Program And Noun Phrase Parser For Unrestricted Text，22
Bleu: A Method For Automatic Evaluation Of Machine Translation，1272	A Stochastic Parts Program And Noun Phrase Parser For Unrestricted Text，22	Finding Clauses In Unrestricted Text By Finitary And Stochastic Methods，1
The Mathematics Of Statistical Machine Translation: Parameter Estimation，298	The Mathematics Of Statistical Machine Translation: Parameter Estimation，298	The Contribution Of Parsing To Prosodic Phrasing In An Experimental Text-To-Speech System，unknown

- venue：三列分别为$\alpha=0.1,0.5,0.9$时，算法得到top3的venue名称，可见α对结果影响很小。

$\alpha=0.1$	$\alpha=0.5$	$\alpha=0.9$

ACL	ACL	ACL
CL	CL	CL
COLING	COLING	COLING

五、一些细节

1. 对引用关系图$\boldsymbol{P}$的处理：在实际中$\boldsymbol{P}$经常是一个稀疏矩阵，如果在编程中建立一个$|\boldsymbol{V}| \times |\boldsymbol{V}|$的二维数组来存储$\boldsymbol{P}$，大量的空间会存为0，这样过于浪费空间，甚至在$|\boldsymbol{V}|$较大时出现内存不够用的情况。于是，我利用索引来存储$\boldsymbol{P}$。以author的引用关系图为例，我建立字典authorLink(key=$\boldsymbol{a}$, value=$\boldsymbol{L_a}$)，$\boldsymbol{a}$为作者名字，$\boldsymbol{L_a}$是一个列表，其中记录了$\boldsymbol{a}$引用的其它作者。
2. 为什么选用noself：ANN数据集中有两类数据，一种记录了自己对自己的引用，另一种是noself，没有记录自己对自己的引用，我选择了后一种数据来进行PageRank。这样的选择与实际场景有关，如果利用第一种数据，一个作者就可以不断引用自己的文章，这样就可以拉高自己文章的PageRank值，成为一种恶意“刷票”。第二种数据规避了这种情况。
3. PageRank收敛的标志：PageRank需要不断迭代，在具体编程中，我需要知道算法何时收敛，从而停止迭代。我的方案是计算上一轮迭代和本轮迭代中各个节点的PageRank差值的绝对值，即$\sum_{i=1}^{|\boldsymbol{V}|} |\boldsymbol{pg_i^k} - \boldsymbol{pg_i^{k-1}}|$，如果小于$\boldsymbol{1e^{-7}}$，则认为PageRank收敛，停止迭代。
4. venue引用关系图中边权重的确定：首先，venue引用关系图上的权重是不同的。比如A会议引用了B会议100篇文章、C会议5篇文章，那么在A看来，显然B的重要程度高于C。为了体现这一点，我把引用关系图上的边权设置成paper的引用量。
5. 原始数据中的bug：`acl-meta.txt`中应该每个条目占每6行数据（分别是id，author，title，venue，year和一个空行），但是id=J11-2003的文章title占用了两行，需要特殊处理。