2023数据库设计与实践 实验报告

项目名：利用HITS算法分析流行短语和网站

姓名：陈家豪

学号：19307130210

一、项目简介

本项目的主要目的是利用HITS算法分析某一段时间内网络中最流行的短语以及最流行的网站。

项目的数据集来自于斯坦福大学的**Memetracker**项目，该项目通过分析来自100万个网络资源(从大众媒体到个人博客)每天的新闻报道和博客文章，跟踪范围内出现最频繁的短语。这使得我们可以分析每天有哪些不同的事件在竞争新闻和博客头条。项目调查的时间为2008年美国总统大选期间，项目统计了新闻和博客中出现的前50条热词的频率。

二、算法

本项目采用的算法是HITS（Hyperlink-Induced Topic Search）算法，该算法在1999年由Jon Kleinberg 提出，原论文为《Authoritative Sources in a Hyperlinked Environment》。该方法最初用于分析网络中的权威信息来源。

2.1 算法原理

在HITS算法中，网页被分为两种：hub页面和authority页面。hub页面是指包含了很多指向authority页面的链接的网页，比如国内的一些门户网站，比如博客、新闻网站等；authority页面则指那些包含有实质性内容的网页。HITS算法的最初目的是：当用户查询时，返回给用户高质量的authority页面。

HITS算法基于下面两个假设：1）一个高质量的authority页面会被很多高质量的hub页面所指向。2）一个高质量的hub页面会指向很多高质量的authority页面。

而一个页面的“质量”，由这个页面的hub值和authority值确定。其确定方法为：

页面hub值等于所有它指向的页面的authority值之和。

页面authority值等于所有指向它的页面的hub值之和。

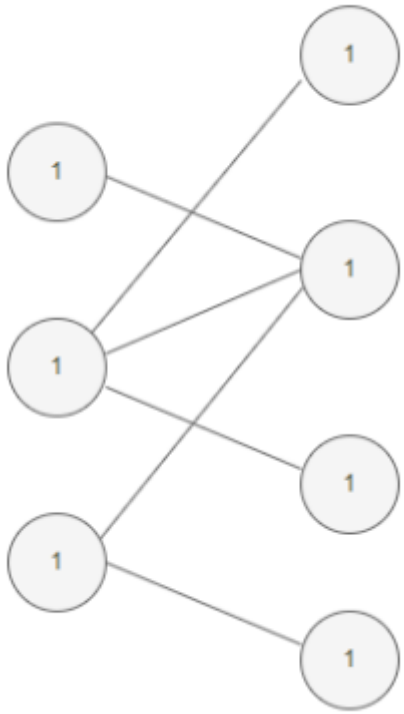
我们在开始时将所有页面的hub,authority值置为1，并通过以上两个步骤不断迭代页面的hub,authority值，则最终会得到收敛的值。

最终我们选取hub值和authority值最大的网站、网页。

2.2 算法流程演示

首先我们需要将数据集转换成一个二分图。hub节点集和authority节点集分别为二分图的两个节点集，如果一个hub节点引用了某authority节点，则在它们之间建立一条边。我们用矩阵来表示边的关系。

假设有如下节点：



则矩阵表示为：

$$A=\begin{matrix}0&1&0&0\\1&1&1&0\\0&1&0&1\end{matrix}$$

hub向量为：$\begin{pmatrix}1\\1\\1\end{pmatrix}$，标准化后为 $\begin{pmatrix}\frac{1}{\sqrt{3}}\\\frac{1}{\sqrt{3}}\\\frac{1}{\sqrt{3}}\end{pmatrix}$，authority向量为：$\begin{pmatrix}1\\1\\1\\1\end{pmatrix}$，标准化 $\begin{pmatrix}\frac{1}{2}\\\frac{1}{2}\\\frac{1}{2}\\\frac{1}{2}\end{pmatrix}$

第一轮迭代后，

$$\text{authority}=A^{T}\cdot\text{hub}=(\frac{1}{\sqrt{3}},\sqrt{3},\frac{1}{\sqrt{3}},\frac{1}{\sqrt{3}})=(\frac{1}{2\sqrt{3}},\frac{\sqrt{3}}{2},\frac{1}{2\sqrt{3}},\frac{1}{2\sqrt{3}})$$

$$\text{hub}=A\cdot\text{authority}=(\frac{\sqrt{3}}{2},\frac{5\sqrt{3}}{6},\frac{2\sqrt{3}}{3})=(\frac{3\sqrt{2}}{10},\frac{\sqrt{2}}{2},\frac{2\sqrt{2}}{5})$$

经过多轮迭代，我们可以得到一个收敛的值。

用python脚本模拟结果如下：

```
matrix = np.array([[0, 1, 0, 0], [1, 1, 1, 0], [0, 1, 0, 1]])
hub = standard(np.ones(3))
auth = standard(np.ones(4))

iter_k = 10
for i in range(iter_k):
    auth, hub = iterate(hub, matrix)
    print("iter:", i)
    print("最热短语：", np.argmax(auth))
    print(auth.max())
    print("最热网站", np.argmax(hub))
    print(hub.max())
```

在迭代第5次时即收敛

```
iter: 5
最热短语： 1
0.8152271848785877
最热网站 1
0.7557861203525478
```

三、数据集

数据集简介

源数据集为`database.sqlite`，在网站`[SNAP Memetracker | Kaggle](https://www.kaggle.com/datasets/snap/snap-memetracker?resource=download)`下载

数据库大小为2.82GB， 包含3个table: `articles` `links` `quotes`

`articles`：字段有`article_id`,`date`,`url`记录了收集的所有文章，以及其来源网址。total_rows : 4542920

`quotes`：字段有`article_id`,`phrase`，记录了各个文章中所引用的热门短语。total_rows : 7956125

数据集处理

源数据集不便于直接利用，为了提高分析效率，我们将对数据集进行整理。整理脚本在`data.py`中。

由于分析最热门的文章可能没有太大意义，我们希望分析的是最热门的网址，所以应该建立起网站与热词直接的直接联系。

首先，我们遍历`articles`表，将所有出现过的url进行域名抽取，用set类型保留不重复域名，存入`domains.jsonl`中。

```
{"id": 0, "domain": "whitenoiseinsanity.wordpress.com"}
{"id": 1, "domain": "asiantsblog.com"}
{"id": 2, "domain": "baracuteycubano.blogspot.com"}
```

遍历 `quotes` 表，用set类型获取所有不重复的phrase，存入`phrase.jsonl`中。

```
{"id": 0, "phrase": "dies ist eine der wenigen gelegenheiten die wir haben die
spieler selbst treffen zu k nnen und wir k nnen es kaum erwarten die ver
ffentlichung von wrath of the lich king mit ihnen zu feiern"}
{"id": 1, "phrase": "voi che ntendendo"}
{"id": 2, "phrase": "we know what is out there ahead of us we can still achieve our
goals"}
```

然后我们将建立HITS算法所需要的3个数据：向量hub, auth, 矩阵M

构建向量和矩阵的脚本在 `graph.py` 中

首先，hub,auth向量初始就是两个单位向量，维度就是`domains.jsonl`， `phrase.jsonl`中的数据的个数，设为a,b。M矩阵是一个a*b大小的矩阵，初始时都为0，如果id为a的domain与id为b的phrase有联系，则M[a, b]为1.

然后从`domains.jsonl`， `phrase.jsonl`两个文件构建两个map，key为domain或phrase， value为各自id。此处的目的是在后面构建矩阵的过程中减少检索的时间。同时也为article表建立索引，可以根据article_id快速检索到域名。

然后遍历quotes表中的每一项。每一行中有article, phrase之间的关联，可以根据之前的map和索引快速建立起domain和phrase之间的关联，将矩阵相应位置置为1。代码如下：

```
def get_graph():
    domain_map = load_domain()
    phrase_map = load_phrase()
    article_domain_list = load_articles()

    hub_size = len(domain_map)
    auth_size = len(phrase_map)

    matrix = np.zeros((hub_size, auth_size))

    f = jsonlines.open('quotes.jsonl', 'r')
    for line in f:
        article_id = line['article_id']
        phrase = line['phrase']
        phrase_id = phrase_map[phrase]

        domain = article_domain_list[article_id]
        domain_id = domain_map[domain]

        matrix[domain_id, phrase_id] = 1
```

```
    return matrix
```

此时我们已经准备好了算法所需的数据：hub, auth, matrix。

四、代码运算结果

算法运行脚本在`main.py`中。

设置迭代次数为k。每次迭代的过程为：

向量标准化->auth = M^T · hub , hub = M · auth

根据向量中最大值的下标，查询得最热门网站域名和短语，打印结果。

```
矩阵大小： 23083 x 104084
iter: 0
最热短语： joe the plumber
0.1580732833716714
最热网站 us.rd.yahoo.com
0.4080976038426851
iter: 1
最热短语： saturday night live
0.09094283577390083
最热网站 blog.myspace.com
0.640314909168497
iter: 2
最热短语： saturday night live
0.061878493758292547
最热网站 blog.myspace.com
0.8239919368477467
iter: 3
最热短语： saturday night live
0.046537088601320165
最热网站 blog.myspace.com
0.9244738075538651
iter: 4
最热短语： saturday night live
0.03623602061568054
最热网站 blog.myspace.com
0.9686235907333696
iter: 5
最热短语： saturday night live
0.029593933191350584
最热网站 blog.myspace.com
0.9862562138017548
iter: 6
最热短语： saturday night live
0.02552436144996694
最热网站 blog.myspace.com
```

```
0.9932430335659284
iter: 7
最热短语： saturday night live
0.0230942815773655
最热网站 blog.myspace.com
0.9961212475624179
iter: 8
最热短语： saturday night live
0.021659600820556454
最热网站 blog.myspace.com
0.9973826849197525
iter: 9
最热短语： saturday night live
0.02081691884768199
最热网站 blog.myspace.com
0.9979750954996259
```

过程中出现的热词，joe the plumber，是2008美国大选时的一个名人；saturday night live是当时美国热门的一个电视节目。

网站blog.myspace.com，是当时最流行的社交网站之一，这符合我们的预期。

五、结论

HITS算法通过网站与热词直接的联系，分析出了2008年间最流行的网站blog.myspace.com，以及最流行的热词saturday night live，具有一定的实际意义。