Web信息处理与应用

第六节

排 序

徐童

2021.10.18

上节回顾

- **更一般的检索方式**

- 采用**排序方式**代替严格匹配模式
  - 在排序检索中，系统根据文档与查询的**相关性排序**返回文档集合中的合适文档，而不是简单匹配查询条件
  - 自由文本查询：用户查询条件是**自然语言描述**，而不是由查询词项构造的表达式。
- 当系统给出的是**有序**的查询结果时，结果数目将不再是个问题
  - 着眼于给出Top N结果，而不是完整结果

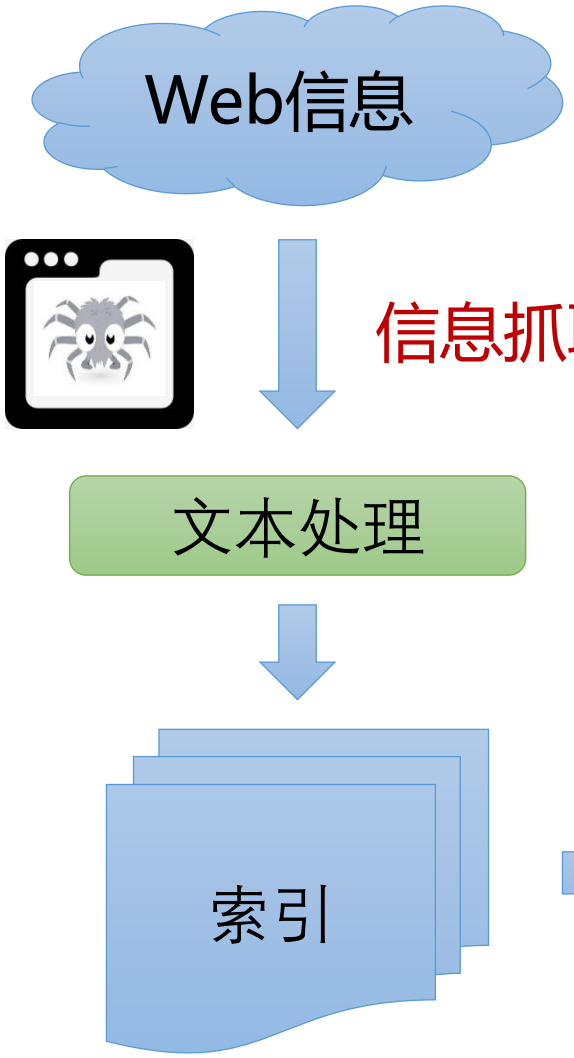
上节回顾

- **相关性反馈与查询优化**

- 用户在查询后标记相关/不相关，然后迭代更新查询，以获得更好的结果
- 相关性反馈的动机
  - 你也许无法表达想要找的内容，但是你至少能够判断所看到的内容
    - “为我提供更多*相似的文档*……”
  - **基于相关性反馈的迭代过程，本质上是网页排序不断优化的过程。**

本节主题

- 本课程所要解决的问题



第五个问题：

如何对网页进行排序，使得用户能够最快获得所需信息？

- **排序问题的难点何在?**

- 传统信息检索方法的两个重要内在假设
  - 被索引的信息具有很高且同等质量，至少在信息的组织和内容上可靠。
  - 信息检索的用户应具有一定的相关知识和技能。
    - 更类似于数字图书馆或专业搜索引擎，e.g., 知网
- 然而，Web Search的现实与假设有巨大落差
  - Web网页的信息组织与内容质量参差不齐
  - 用户庞杂且缺乏知识和经验
  - 用户意图多样，存在巨大差异

- **不同搜索引擎，由于数据、侧重不同，所得结果也不同**

- **不同搜索引擎，由于数据、侧重不同，所得结果也不同**

- **一个好的排序应该是怎样?**

- 如果得了牙疼，应该去看哪个医生呢?
  - A医生，既治眼病，又治胃病
  - B医生，既治牙病，又治眼病，还治胃病
  - C医生，专治牙病
- 按照布尔检索的思路，B和C都满足需求。
- 如果从医生专长的考虑，C是更好的选择。

- **一个好的排序应该是怎样？（续）**

- 如果我们有更丰富的信息？
  - B医生，从医二十年，经验丰富
  - C医生，只有五年从医经验
- 从择医的角度考虑，需要同时考虑专长与医术
- 对于**网页排序**而言，也是如此
  - 网页内容匹配程度 → 医生的专长
  - 网页内容的质量 → 医生的经验与水平

本节目录

- 信息检索与相关度计算
  - **信息检索模型概述**
  - 向量空间模型
- 网页权威性计算
- 排序学习问题

信息检索模型概述

- 信息检索模型是用来描述文档与查询的表示形式与相关性的框架
  - 信息检索的实质是对文档基于相关性进行排序
  - 好的信息检索模型，可以在理解用户的基础之上，产生近似用户决策的结果，从而在顶部返回最相关的信息
- 信息检索模型的形式化表述： [D, Q, F, R(Di, q)]
  - D：文档表达
  - Q：查询表达
  - F：查询与文档间的匹配框架
  - R：查询与文档间的相关性度量函数（Di与q分别表示特定文档与查询）

- **基本信息检索模型：布尔模型**

- 最早的信息检索模型
  - 1957年，布尔逻辑是否可能应用于计算机信息检索已引起了讨论
- 以布尔模型为例，对应介绍信息检索模型的形式化表述
  - D：文档表达——词项的组合（注意是词项不是单词！）
  - Q：查询表达——布尔表达式（词项 + 布尔运算符）
  - F：完全匹配（二值匹配）
  - R：满足布尔表达式，相关性为1，否则为0（即使部分满足）

本节目录

- **信息检索与相关度计算**
  - 信息检索模型概述
  - **向量空间模型**
- 网页权威性计算
- 排序学习问题

- **从匹配到相关性排序**
  - 布尔模型的重要局限性之一在于二值化匹配
    - 非0即1，无法实现相关性排序
  - 一个好的信息检索系统，应该满足如下条件：
    - 相关度高的文档，应该比相关度低的文档排名更高
      - 所以要有区分！不能吃大锅饭
      - 如何实现？对每个【查询-文档】赋予[0,1]之间的一个分值
        - 该分值度量了文档与查询之间的匹配程度

- **量化方法（1）：Jaccard系数**

  - 布尔模型的一个假设：所有的查询条件都必须满足
    - 假如打破这一约束，能够实现相关性的[0,1]量化?
  - 1901年，Paul Jaccard提出计算两个集合重合度的方法
    - 如果A与B为两个集合，那么有：

$$\mathrm{JACCARD}(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Paul Jaccard (1868-1944)

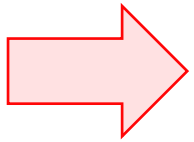
    - 显然，我们有： $\mathrm{JACCARD}(A, A)=1$， $\mathrm{JACCARD}(A, B)=0$（如果$A \cap B = \emptyset$）

- **量化方法（1）：Jaccard系数**

  - 由此，我们得到了一个描述集合重合度的，处于[0,1]之间的值
    - 注意前提：“集合”，即仍将文档视作词项的集合
  - 通过计算查询与文档之间的重合度，可以初步估算文档的相关性

查询 **“ides of March”**

文档 **“Caesar died in March”**



$$\text{JACCARD}(q,d) = 1/6$$

- **量化方法（1）：Jaccard系数**

  - 然而，Jaccard系数显然仍过于简单、粗糙
    - 从完全匹配到部分匹配是一种进步，但查询词本身未做重要性区分
    - 不考虑词项频率，即词项在文档中的出现次数
    - 没有仔细考虑文档的长度因素
    - 罕见词比高频词的信息量更大，Jaccard系数没有考虑这个信息

- **量化方法（2）：词项频率TF**

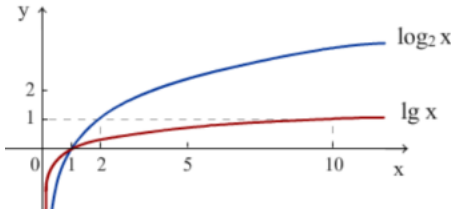
  - 比Jaccard更进一步，如何通过词项区分文档权重?
  - 一个直观的想法：查询词在文档中出现得越多，该文档越相关
    - 词项频率TF(t,d)，指词项 t 在文档 d 中出现的次数（Term Frequency）
    - 利用原始的TF值，可以粗略计算文档的相关性
      - 但存在一定问题，相关性与频率并不线性相关
        - 某个词在文档A中出现100次，在文档B中出现10次，A比B相关10倍?

- **量化方法（2）：词项频率TF**

  - 原始TF基础之上的改进：引入对数词频

$$wf_{t,d} = \begin{cases} 1 + \log_{10} tf_{t,d} & tf_{t,d} > 0 \\ 0 & otherwise \end{cases}$$

    - 通过这种方式，数量级的差异性所造成的影响变得更为缓和
  - 由此，可以将文档与词项的匹配得分定义为所有同时出现在查询与文档中的词项其对数词频之和，即$\sum_{t \in q \cap d}(1 + tf_{t,d})$
    - 如果没有公共词项，显然得分为0



- **量化方法（2）：词项频率TF**

- 上上节课的回顾：基础检索中的关联矩阵
  - 给文档建立索引（Index），出现即为0，未出现即为1

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

- **量化方法（2）：词项频率TF**

- 基于词项频率的改进版本实例
  - 此时，不同文档的相关性出现了区分

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

- **量化方法（3）：文档频率DF**

  - 如何再近一步？
    - 有些词在单个文档中出现的多，是因为这个词本身就很常用。
  - 回顾：停用词的概念，为什么要去除停用词？
    - 数字、副词等与语义关系不大的词常作为停用词被处理。
    - 对最后结果的排序没什么贡献，反而可能产生干扰。
    - 特定领域中有其专用的停用词！
      - 如URL中的WWW，Wikipedia中的Wiki

- **量化方法（3）：文档频率DF**

  - 一个什么样的词真正有区分度?

    - 香农（Shannon）有话说：

      香农（C. E. Shannon）信息论

      信息量是指从N个相等可能事件中选出一个事件所需要的信息度量或含量，也就是在辩识N个事件中特定的一个事件的过程中所需要提问“是或否”的最少次数．香农信息论应用概率来描述不确定性。信息是用不确定性的量度定义的。
      一个消息的可能性愈小，其信息量愈多；而消息的可能性愈大，则其信息愈少。事件出现的概率小，不确定性越多，信息量就大，反之则少。

  - 总结起来：罕见词的信息量更为丰富，而频繁词的信息量相对较少

    - 相应的，如果查询中包含某个罕见词，则包含这个词的文档可能很相关

- **量化方法（3）：文档频率DF**

  - 基于上述思想，引入新的度量机制：文档频率（Document Frequency）
    - df_t，指出现词项 t 的文档数量
      - df_t是与词项 t 的成反比的一个值
    - 相应的，我们一般采用逆文档频率（Inverse DF）来衡量

$$idf_t = log_{10} \frac{N}{df_t}$$

      - IDF由剑桥大学的Spock Jones于1972年提出，并由Salton推广
      - 需要注意，这里同样引入对数计算方式来抑制DF中数量级的影响

- **量化方法（3）：文档频率DF**

  - 简单的 idf 计算实例（当 N = 1,000,000时 ）

    - 其中，过于频繁的词（如the）的作用被完全抹掉了

词项	df_t	idf_t
calpurnia	1	6
animal	100	4
sunday	1000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

- **量化方法（4）：集大成的tf-idf**

  - TF与DF，从两个角度为衡量文档相关提供了量化依据
    - 如何将两者整合起来，实现更完整的描述?
    - 乘起来就完事了~

$$W_{t,d} = \left(1 + \log tf_{t,d}\right) \cdot \log \frac{N}{df_t}$$

    - 同时，解答了一个问题：什么样的词适合衡量文档相关性?
      - 在少数文档内多次出现的词
      - 指标随词项频率增大而增大，随词项罕见度增大而增大

- **量化方法（4）：集大成的tf-idf**

- 基于 tf-idf 的再次改进版本实例
  - 此时，不同文档的向量化表达已经呼之欲出

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

- **量化方法（5）：千呼万唤始出来的VSM**

- 从TF到DF再到tf-idf，我们对于文档中的词项有了愈加合理的描述
- 与此同时，相应的文档向量也具有了更丰富的信息量
- 由此，我们终于可以引出向量空间模型（Vector Space Model）

- 回顾的回顾：Salton在SMART系统中提出了著名的 **向量空间模型**
  - 1960年代，康奈尔大学的Gerard Salton研发了SMART系统，被视作信息检索的鼻祖

Gerard Salton (1927-1995)

- **量化方法（5）：千呼万唤始出来的VSM**

- 向量空间模型（Vector Space Model，VSM）
  - 每个文档和查询视作一个词项权重构成的向量
  - 查询时通过比较向量之间相似性来进行匹配
- 如果用开头提到的形式化表述加以概括?
  - D：文档表达，每个文档可视作一个向量，其中每一维对应词项的tf-idf值
  - Q：查询表达，可视作一个向量，其中每一维对应词项的tf-idf值
  - F：非完全匹配方式
  - R：使用两个向量之间的相似度来度量文档与查询之间的相关性

- **量化方法（5）：千呼万唤始出来的VSM**

- 文档表示示例

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

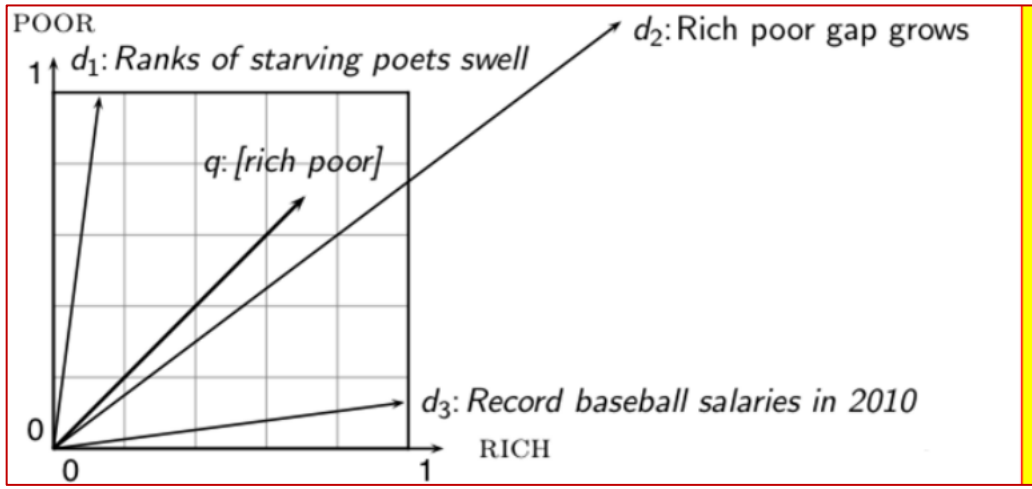
- 所以，D1对应的向量为(5.25, 1.21, 8.59, 0, 2.85, 1.51, 1.37)

- **量化方法（5）：千呼万唤始出来的VSM**

- 如何计算查询与文档之间的相似度?
  - 最基本的方法：欧氏距离

$$dist(\vec{x},\vec{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_n - y_n)^2}$$

  - 然而，欧氏距离并不是一个好的选择，它对于向量长度（文档长度）非常敏感



尽管查询*q*和文档d_2的词项分布非常相似，但是采用欧氏距离计算它们对应向量之间的距离非常大

- **量化方法（5）：千呼万唤始出来的VSM**

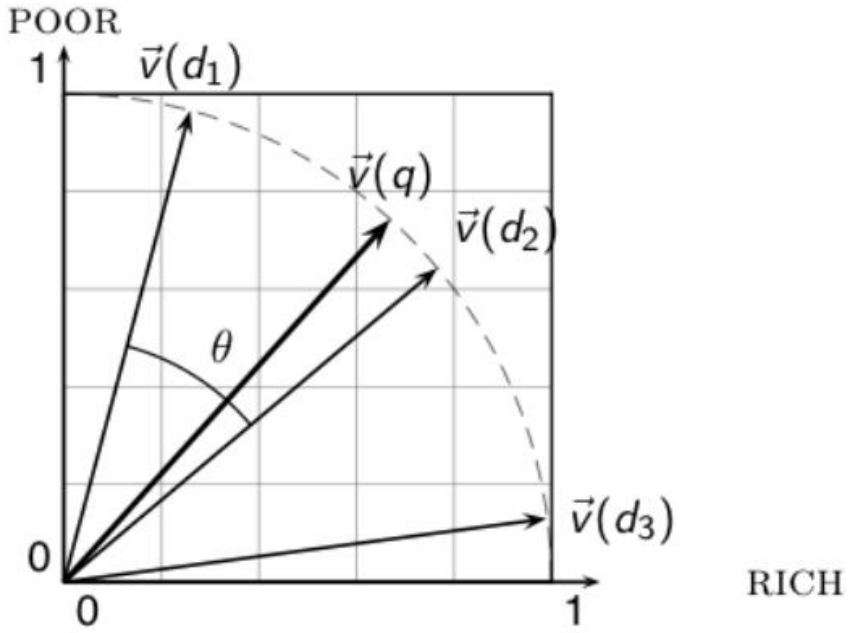
- 事实上，类似欧氏距离缺陷这样的问题，在许多应用场景都会出现
  - 例如，豆瓣电影评分中某用户打分均值或方差的影响
  - 假如有某部电影，三个用户的打分分别如下：
    - A：10，8，9
    - B：4，2，3
    - C：8，10，9
    - 哪两个用户的偏好更为一致？

- **量化方法（5）：千呼万唤始出来的VSM**

- 更为合适的方案：余弦相似度
  - 按照文档向量与查询向量的夹角大小来计算

$$\cos\left(\vec{q},\vec{d}\right)=\frac{\sum_{i=1}^{|V|}q_i d_i}{\sqrt{\sum_{i=1}^{|V|}q_i^2}\sqrt{\sum_{i=1}^{|V|}d_i^2}}$$

  - 显然，向量越一致，夹角越小，cosine值越高
    - 相应的，它们之间的相似度也就越高
    - 即使这种情况，它们的欧氏距离可能很大



- **量化方法（5）：千呼万唤始出来的VSM**

- 余弦相似度的计算实例
  - 假如有两个文档和一个查询
    - D1 = (0.5, 0.8, 0.3)
    - D2 = (0.9, 0.4, 0.2)
    - Q = (1.5, 1.0, 0)

$$Cosine(D_1, Q) = \frac{(0.5 \times 1.5) + (0.8 \times 1.0)}{\sqrt{(0.5^2 + 0.8^2 + 0.3^2)(1.5^2 + 1.0^2)}} = \frac{1.55}{\sqrt{(0.98 \times 3.25)}} = 0.87$$

$$Cosine(D_2, Q) = \frac{(0.9 \times 1.5) + (0.4 \times 1.0)}{\sqrt{(0.9^2 + 0.4^2 + 0.2^2)(1.5^2 + 1.0^2)}} = \frac{1.75}{\sqrt{(1.01 \times 3.25)}} = 0.97$$

- **量化方法（5）：千呼万唤始出来的VSM**

- 向量空间模型的总结
  - 首先，将文档与查询表示成词项的tf-idf权重向量（也可采用其他方法）
  - 其次，计算两个向量之间的某种相似度（如余弦相似度）
  - 最后，按相似度大小进行排序，将Top-K的文档返回给用户
- 优点：简洁直观，可以支持多种不同度量或权重方式，实用效果不错
- 缺点：缺乏语义层面的理解和匹配，同时依赖tf-idf值也可能造成干扰
  - 用户无法描述词项之间的关系，词项之间的独立性假设实际上不成立
    - 例如：贝利 + 足球，郎平 + 排球

本节目录

- 信息检索与相关度计算
- **网页权威性计算**
  - **PageRank及其衍生模型**
  - HITS算法与网页角色区分
- 排序学习问题

- 再次回顾：从Web1.0到Web2.0

Baidu 百度

Google

腾讯新闻
NEWS.QQ.COM

头条
今日头条
你关心的 才是头条

信息提供者

信息整合者

信息消费者

- 再次回顾：从Web1.0到Web2.0

twitter

新浪微博 weibo.com

YouTube

bilibili

我们每个人，既是信息的生产者，也是信息的消费者

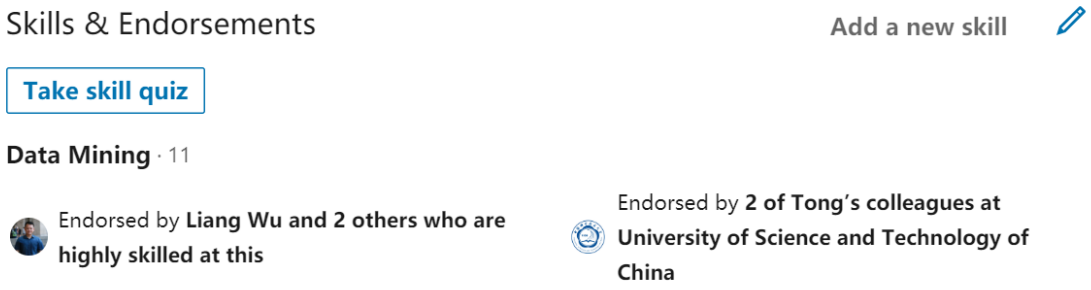
- **Web 2.0时代的隐患与PageRank的价值**

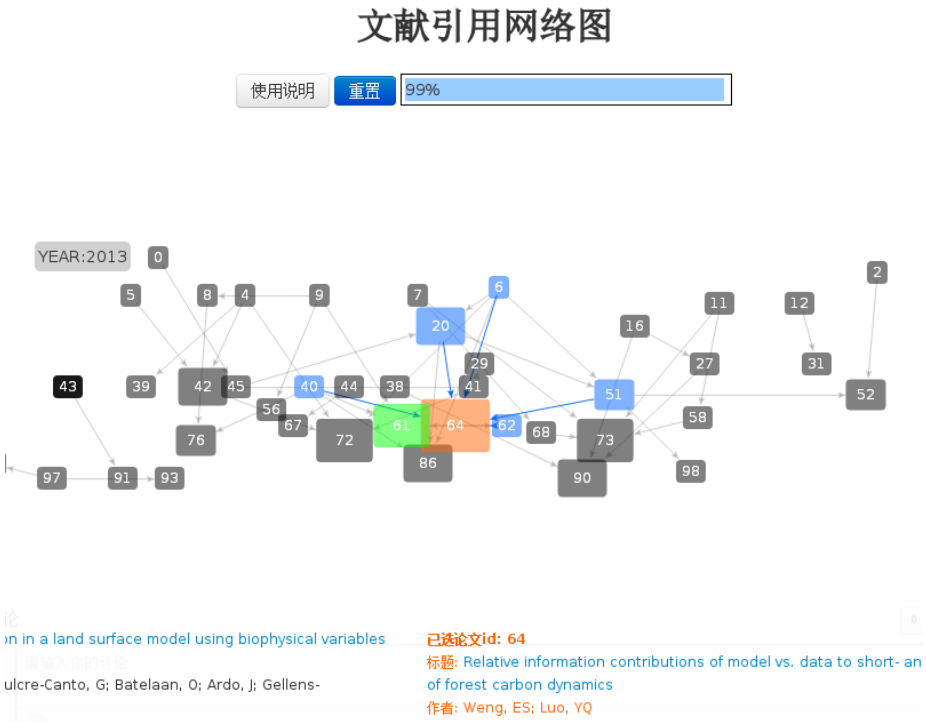
- 大众创造内容的时代后遗症：内容权威性的下降
  - 信息的错漏、谣言与虚假信息的兴起
  - 低质内容服务商大量涌现
- 如何在衡量相关性的同时，确保内容权威性?

PageRank

- **一个启发式的想法**

- 大家说好才是真的好：来自同行的endorsement
  - 区分“点赞”的人将获得更好区分效果





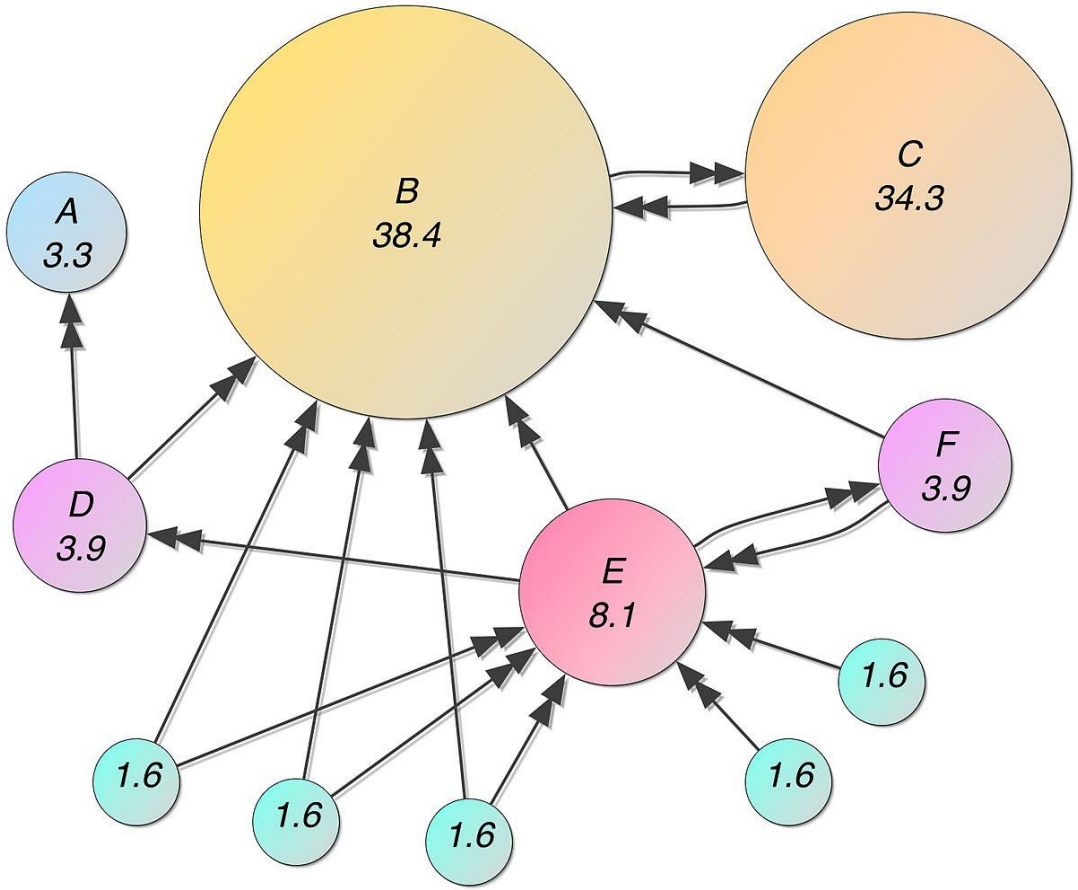
PageRank

- **PageRank的历史**

- Sergey Brin 和Lawrence Page 在1998年提出了PageRank 算法
  - Lawrence Page, Sergey Brin, Rajeev Motwani, Terry Winograd，*The PageRank Citation Ranking: Bringing Order to the Web*，Stanford InfoLab, 1999
    - 截止2020.10.28，引用14148次
    - http://www-db.stanford.edu/~backrub/pageranksub.ps
  - Sergey Brin, Lawrence Page, The Anatomy of a Large-scale Hypertextual Web Search Engine. WWW'98, Computer Networks 30(1-7): 107-117 (1998)
    - 截止2020.10.28，引用20159次
    - https://ai.google/research/pubs/pub334.pdf

- **PageRank的核心思想**

- 将网页（或文档）视作一个点，网页之间的超链接视作一条边，从而形成一个巨大的有向图
  - 点之间的有向连接表示了网页之间相互引用，相互推荐的关系
  - 入度越多，则被引用或推荐的次数越多，网页的重要性就越大



- **PageRank的计算方法**

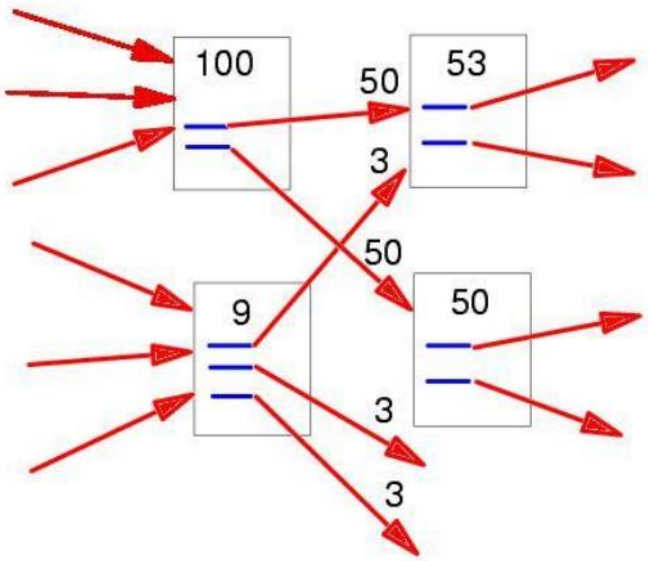
- PageRank的核心公式如下：

$$\mathrm{PR}(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)}$$

  - 其中：
    - PR(pi)为网页pi的PageRank值
    - PR(pj)为指向网页pi的某个网页pj的PageRank值
    - L(pj)为网页pj发出的链接数量
    - d为阻尼系数，取值在0-1之间
    - N为网页总数，M(pi)为链入pi的页面集合

• PageRank的设计思路

- PageRank背后的思想：
  - 总体思路：优质网页引用或推荐的网页，一定还是优质网页
  - 三重衡量标准：
    - 链入链接数：单纯意义上的受欢迎程度
    - 链入链接本身是否推荐度高：欢迎本身是否具有权威性
    - 链入链接源页面的链接数：被选中的几率
      - 体现源网页是否滥发推荐

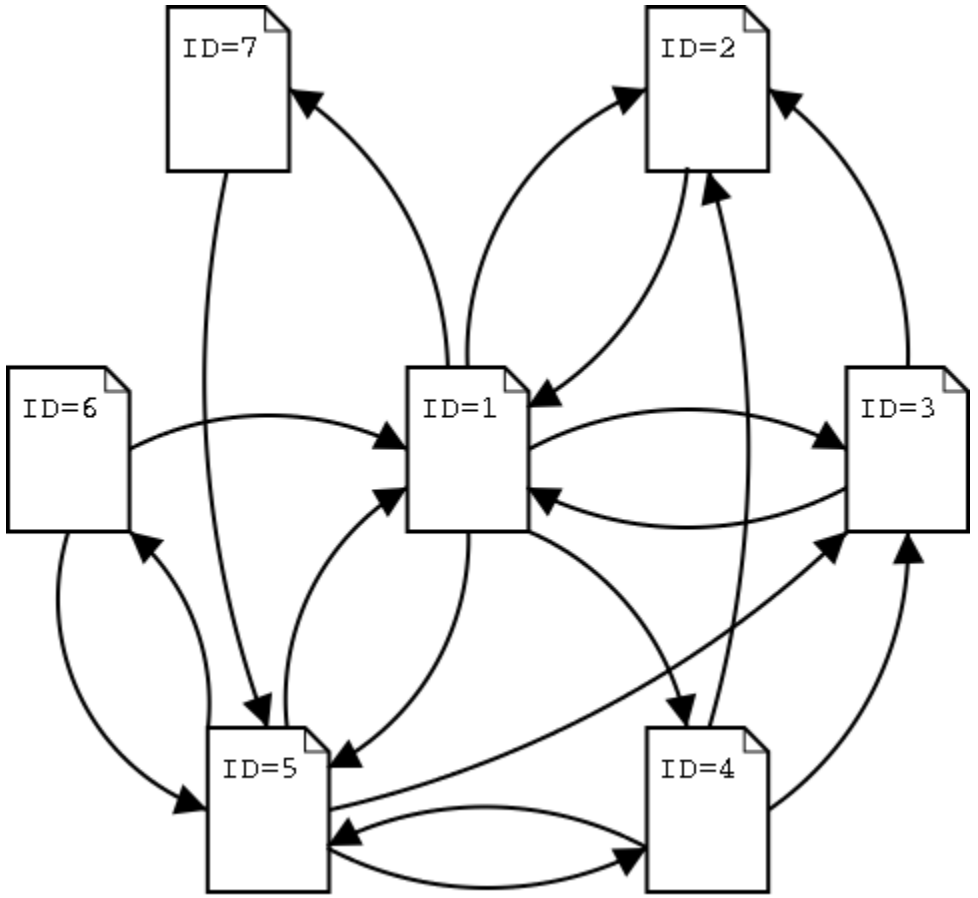


- **PageRank的实现过程**

- PageRank的迭代计算过程：
  - 采用近似迭代算法计算PageRank值
    - 首先给每个网页赋予一个初值，例如1/N
    - 然后，利用之前的公式进行迭代有限次计算，得到近似结果
- 先前提到的论文显示，实际进行大约100次迭代才能得到整个网络的PageRank。
- 中等规模的网站（如论文中提到的26M个网站），需要数小时完成这一过程

- **PageRank的计算实例**

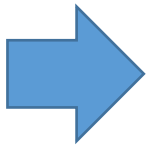
链接源I D	链接目标 ID
1	2,3 ,4,5, 7
2	1
3	1,2
4	2,3,5
5	1,3,4,6
6	1,5
7	5



- **PageRank的计算实例**

- 从邻接矩阵到跳转矩阵
  - 转置后，将各个数值除以非零元素数（即出边数量）

$$A=\begin{bmatrix} 0,1,1,1,1,0,1; \\ 1,0,0,0,0,0,0; \\ 1,1,0,0,0,0,0; \\ 0,1,1,0,1,0,0; \\ 1,0,1,1,0,1,0; \\ 1,0,0,0,1,0,0; \\ 0,0,0,0,1,0,0; \end{bmatrix}$$



$$R=\begin{bmatrix} 0, & 1, & 1/2, & 0, & 1/4, & 1/2, & 0; \\ 1/5, & 0, & 1/2, & 1/3, & 0, & 0, & 0; \\ 1/5, & 0, & 0, & 1/3, & 1/4, & 0, & 0; \\ 1/5, & 0, & 0, & 0, & 1/4, & 0, & 0; \\ 1/5, & 0, & 0, & 1/3, & 0, & 1/2, & 1; \\ 0, & 0, & 0, & 0, & 1/4, & 0, & 0; \\ 1/5, & 0, & 0, & 0, & 0, & 0, & 0; \end{bmatrix}$$

- **PageRank的计算实例**

- 开始计算，假设所有PageRank值初始均为1/N

$$R=\begin{bmatrix} 0, & 1, & 1/2, & 0, & 1/4, & 1/2, & 0; \\ 1/5, & 0, & 1/2, & 1/3, & 0, & 0, & 0; \\ 1/5, & 0, & 0, & 1/3, & 1/4, & 0, & 0; \\ 1/5, & 0, & 0, & 0, & 1/4, & 0, & 0; \\ 1/5, & 0, & 0, & 1/3, & 0, & 1/2, & 1; \\ 0, & 0, & 0, & 0, & 1/4, & 0, & 0; \\ 1/5, & 0, & 0, & 0, & 0, & 0, & 0; \end{bmatrix} + M=\begin{bmatrix} 1/7 \\ 1/7 \\ 1/7 \\ 1/7 \\ 1/7 \\ 1/7 \\ 1/7 \end{bmatrix}$$

以3号节点为例，从先前的图可以看到，3号节点有三条入边，分别来自1、4、5号

同时，1、4、5号节点又各自有5、3、4条出边

根据之前的迭代式我们知道，**假设d=1时**，有R3 = R1 / 5 + R4 / 3 + R5 / 4

这个公式恰好等于上述R矩阵第三行 与 M向量 的内积

- **PageRank的计算实例**

- 因此，我们采用 R·M 来进行迭代计算每个节点的PageRank值

$$R=\begin{bmatrix} 0, & 1, & 1/2, & 0, & 1/4, & 1/2, & 0; \\ 1/5, & 0, & 1/2, & 1/3, & 0, & 0, & 0; \\ 1/5, & 0, & 0, & 1/3, & 1/4, & 0, & 0; \\ 1/5, & 0, & 0, & 0, & 1/4, & 0, & 0; \\ 1/5, & 0, & 0, & 1/3, & 0, & 1/2, & 1; \\ 0, & 0, & 0, & 0, & 1/4, & 0, & 0; \\ 1/5, & 0, & 0, & 0, & 0, & 0, & 0; \end{bmatrix} + M=\begin{bmatrix} 1/7 \\ 1/7 \\ 1/7 \\ 1/7 \\ 1/7 \\ 1/7 \\ 1/7 \end{bmatrix}$$

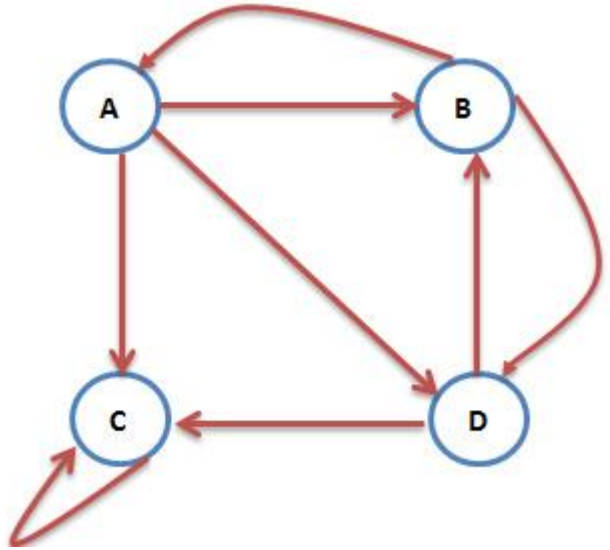
某轮迭代后，若M与M’对应维元素的差值高于阈值则继续迭代，直至收敛

当前M与M’差值过高，继续迭代

$$R \cdot M = \begin{bmatrix} 0.321 \\ 0.148 \\ 0.148 \\ 0.064 \\ 0.148 \\ 0.036 \\ 0.029 \end{bmatrix} = M'$$

- **PageRank中的两类特殊情况**

- PageRank的计算过程，实际上是一个马尔科夫过程
  - 如果计算中出现陷阱节点或终止节点，如何处理？
    - 陷阱节点：只有一条指向自己的边，没有其他出边
    - 终止节点：没有任何出边，如同黑洞
- 此外，孤立节点的存在也会产生一定影响
  - 没有任何入边，单纯使用马尔科夫链无法跳转
    - 仅有初始概率，不再更新，也不影响其他节点

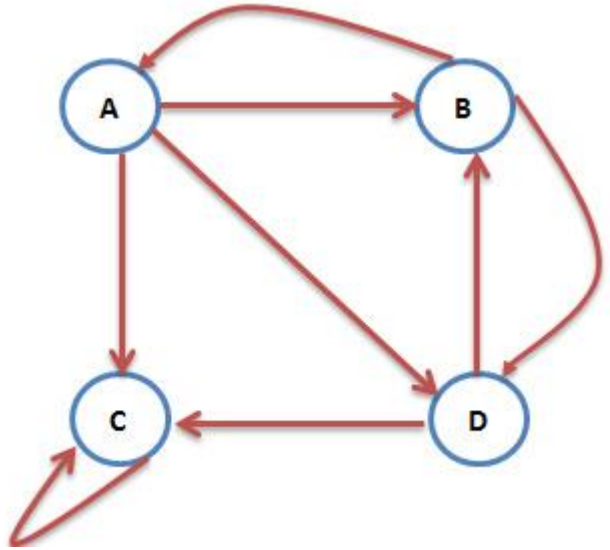


- **PageRank中的两类特殊情况**

- 几类特殊情况的解决：Restart机制
  - 回顾PageRank的公式

$$\mathrm{PR}(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)}$$

  - 其中的(1-d)/N的部分，相当于以一定概率重新选择起点
    - 此时，所有节点以一定等概率被选中作为新起点
    - 由此，跳出了陷阱和黑洞的干扰（所有节点全联通）
    - d一般选择为0.85左右（Google的选择）



- **PageRank中的收敛性问题**

- 如前所述，PageRank的计算过程，实际上是一个马尔科夫过程。

$$\mathrm{PR}(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)}$$

  - 取$A = dM + [(1-d)/N]ee^T$，其中M为跳转矩阵，e为所有元素都为1的列向量
  - 则有$Pn+1 = APn$，形成马尔科夫过程

- 前述论文已从理论上证明，不论初始PageRank值如何选取，这种算法都保证了网页排名的估计值能收敛到他们的真实值。如何保证?

- **PageRank中的收敛性问题**

- 马尔科夫过程的三个收敛条件：
  - 转移矩阵A为马尔科夫矩阵
    - A矩阵所有元素都大于等于0，并且每一列的元素和都为1
  - 转移矩阵A为不可约的
    - 当图是强连通时，A为不可约，而Restart保障了这一条件
  - 转移矩阵A为非周期的
- 这三个条件，PageRank算法都满足，因此保障了其收敛性。

PageRank的拓展模型：个性化PageRank

- PageRank在衡量网页的整体权威性方面具有显著的意义
- 然而，用户的个性化因素却没有在PageRank中得以体现
  - 每个用户有自己特定的偏好，绝大多数网页并不会被用户浏览
  - 一种解决方案是将用户偏好排序与PageRank值结合起来
    - 从用户喜爱的网页中挑选更权威的，较为简便，但可能效果一般
  - 另一种方案，模拟用户自己的网页浏览行为，从而计算个性化权威
    - 个性化PageRank（Personalized PageRank）

PageRank的拓展模型：个性化PageRank

- 早在两位大神1999年的论文中，就提到了Personalized PageRank的思想
- 在个性化PageRank中，用户视作网络中的一个虚拟节点
  - 首先，任何跳转行为均从用户开始，即用户自身作为浏览行为的起点
  - 其次，从用户出发的跳转概率不一定视作均等（如1/N）
    - 可根据用户对不同网页的偏好决定其概率
    - 基本公式形如 $\boldsymbol{r} = (1-d)\boldsymbol{M}\boldsymbol{r} + d\boldsymbol{v}$ 的形式
      - v 体现了用户偏好，可视作不同网站在用户偏好下的点击概率

PageRank的拓展模型：主题敏感PageRank

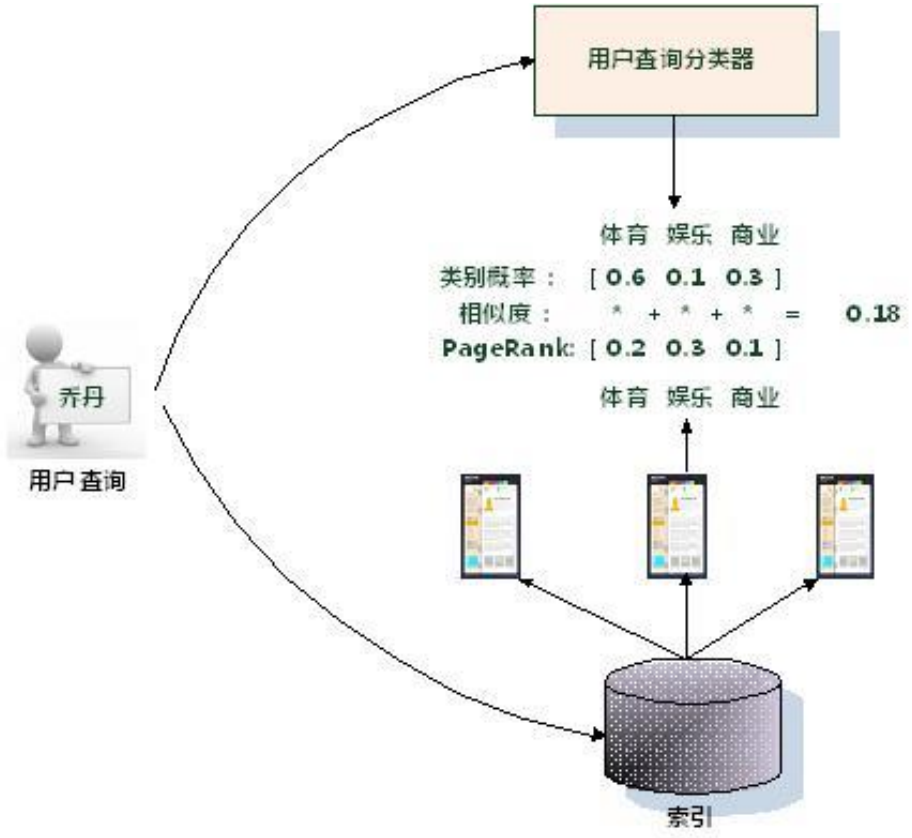
- 个性化PageRank固然体现了用户偏好，然而代价过于巨大
  - 为每一个用户单独计算其PageRank值事实上是个Mission Impossible
- 如何既在一定程度上考虑偏好因素，又减轻计算资源的负担？
  - 一个折中的方案是，以主题为中介，为特定的主题计算相应的PageRank
    - TH Haveliwala, Topic-sensitive pagerank, WWW‘02，已引用3247次
    - 某种意义上，这一模型体现了“仅有同行的评价才有价值”的思想
      - 隶属不同主题的网站，其相互引用和推荐的作用将被削弱

- **PageRank的拓展模型：主题敏感PageRank**

- Topic-sensitive PageRank与基本PageRank的区别
  - 首先，每个网页隶属于某个特定主题
    - 在论文中，作者参考Open Directory Project划分了16个类别
  - 其次，在计算部分，区别主要在于起始节点和Restart部分
    - 对于某个Topic来说，起点仅限于隶属于该Topic的网站
    - (1-d)e/N更改为了(1-d)s/|s|，s为一个N维向量
      - 如果某个网站隶属于该主题，则 s 中的该维为1，反之为0
      - |s| 表示 s 中为 1 的元素的个数。

• PageRank的拓展模型：主题敏感PageRank

- Topic-sensitive PageRank的输出
  - 最终，每个网站将得到一个Topic-sensitive的向量。
    - 即使不属于某个Topic，网页也可以通过PageRank获得相应的数值。
  - 同样，用户的需求也将表示为Topic向量
  - 两个向量的内积可以反应这一网站在这种主题需求下的权威性。

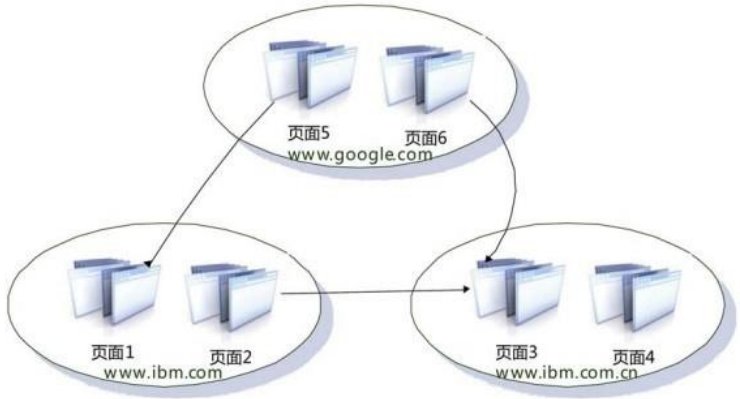


• PageRank的拓展模型：Hilltop算法

- 如前所述，主题敏感PageRank体现了“同行的评价更有价值”的思想
- 然而，即使在这种约束下，PageRank仍有被滥用的危险
  - 过分依赖PageRank衡量网页权威性，可能会鼓励那些试图通过增加大量无效链接而提高PageRank值的作弊手段。
- 如何保障这一指标的可靠性?
  - 一种启发式方法：预先选定一批“高质量网页”，然后通过这批网页发出的链接，来判断其他网页的价值
  - Hilltop算法，由Krishna Baharat 在2000年研究，于2001年申请专利

- **PageRank的拓展模型：Hilltop算法**

- Hilltop算法的基本概念之一：非从属组织网页
  - 满足以下两种情况，将被视作从属组织网页
    - 主机IP地址的前三个字段相同，如182.61.200.X （百度）
    - URL中的主域名段相同，如 XXX.ustc.edu.cn



• PageRank的拓展模型：Hilltop算法

- Hilltop算法的基本概念之二：专家页面
  - 所谓专家页面，即与某个主题相关的高质量页面
  - 专家页面一般通过如下启发式规则进行挑选
    - 至少包括K个出边，K由人为指定
    - K个出边指向的所有页面相互之间的关系都符合“非从属组织页面”
  - 专家页面之外的部分被称作“目标页面集合”
    - 即需要进行PageRank计算，判定其权威性的部分

- **PageRank的拓展模型：Hilltop算法**

- Hilltop算法的基本流程
  - 首先，根据用户的查询需求，选定相关的专家页面，并计算其相关性
  - 其次，遵循PageRank算法，对目标页面集合进行排序
    - 即将专家页面的相关性得分，传递到其他的目标页面上去
  - 最后，整合最相关的专家页面和得分较高的目标页面返回给用户
- 该方法能够一定程度上避免恶意添加链接的作弊行为，同时保证搜索与主题的相关性。但结果严重依赖专家页面的搜索和确定，存在一定偏见。

PageRank的总结

- PageRank是一种基于链接分析的全局网页排序算法
- 优点
  - 对于网页给出全局的重要性排序，并且可以离线完成以保障效率
  - 基础的PageRank算法本身独立于检索主题，可以实现结果的通用
- 缺点
  - 主题无关性，同时对于恶意链接、广告链接等缺乏区分
  - 旧网页得分更高，因为新网页往往少有入链（论文同样如此！）
  - 一般不能单独用于排序，需要与相关性排序方法相结合
  - 效率问题（很多图问题的通病）

本节目录

- 信息检索与相关度计算
- **网页权威性计算**
  - PageRank及其衍生模型
  - **HITS算法与网页角色区分**
- 排序学习问题

- **从PageRank到HITS算法**

- Hilltop算法尝试了网页角色的区分，但仍有进一步的空间
  - 对于基础PageRank算法而言，不同链接的对外输出是平均的
  - 即使通过个性化或主题敏感进行了修正，但对网页的功能仍不区分
    - 例如，专业信息站点与门户网站，在提供信息的种类上区别很大
    - 因此，将对外链接的输出和功能视作等价不符合实际情况

- **从PageRank到HITS算法**

- 在PageRank提出的同一年（1988），康奈尔大学的Jon Kleinberg（又一尊大神）提出了Hyperlink – Induced Topic Search（HITS）
  - Kleinberg博士认为，既然搜索是用户发起的提问，那么页面的重要性衡量就应该遵循用户的检索意图

Kleinberg J M. Authoritative sources in a hyperlinked environment, In Proceedings of the ACM-SIAM symposium on discrete algorithms (SODA 1998). 1998. （引用3138次）

Kleinberg J M. Authoritative sources in a hyperlinked environment[J]. Journal of the ACM (JACM), 1999, 46(5): 604-632. （引用10312次）

- **HITS算法的两个核心概念**

- 权威（Authority）网页与枢纽（Hub）网页的区分
- 权威网页：指某个领域或某个话题相关的高质量网页
  - 如科研领域的中科院之声，视频领域的优酷与爱奇艺等
- 中心网页：类似中介，指向了很多高质量的权威网页
  - 如“hao123”，各个浏览器自带的首页（手动滑稽）
- HITS的目的即在海量网页中找到并区分这些与用户查询主题相关的高质量“Authority”与“Hub”网页，尤其是“Authority”（因为更能满足用户的信息需求）

- **HITS算法的两个基本假设**

- 基本假设与核心概念相互对应
  - 基本假设1：好的Authority会被很多好的Hub指向

$$\forall p, a(p) = \sum_{i=1}^{n} h(i)$$

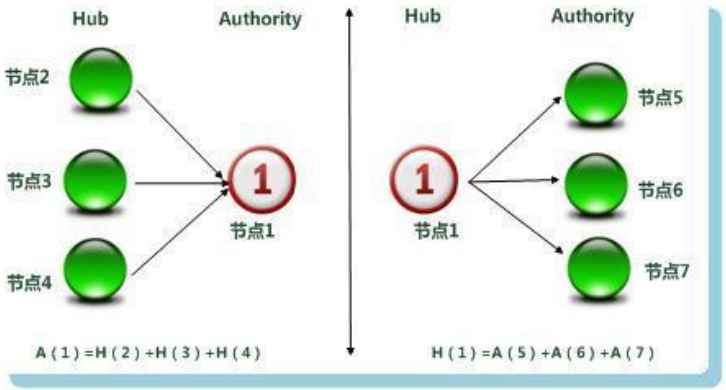
  - 基本假设2：好的Hub会指向很多个好的Authority

$$\forall p, h(p) = \sum_{i=1}^{n} a(i)$$

  - 因此，在HITS算法中，每个网页需要计算两个权威值

- **HITS算法的计算过程**

- 基于先前的基本假设，HITS的计算过程如下：
- 首先，根据关键字获取与查询最相关的少数页面，及与这些页面有链接关系的页面，作为待选集合
- 其次，对所有网页的 a(p) 与 h(p) 进行初始化，可都设为1
- 最后，迭代计算两个步骤，即基本假设所对应的两个公式
  - 重复这一步，直到最终收敛为止
    - 注意每一步中的归一化过程！
  - 输出Authority或Hub值较高的页面



- **HITS算法的计算过程**

- 基于先前的基本假设， HITS的计算过程如下：
  - 假定邻接矩阵为M，Authority向量为a，Hub向量为h
  - 则有如下迭代式：
    - $a_{k+1} = M^T h_k$， $h_{k+1} = M a_{k+1}$
  - 或者，可采用如下迭代式：
    - $a_{k+1} = (M^T M)^k M^T a_0$， $h_{k+1} = (M M^T)^{k+1} h_0$
    - 其中，a_0, h_0为Authority/Hub向量的初始值，可设为全1向量

HITS算法的优缺点

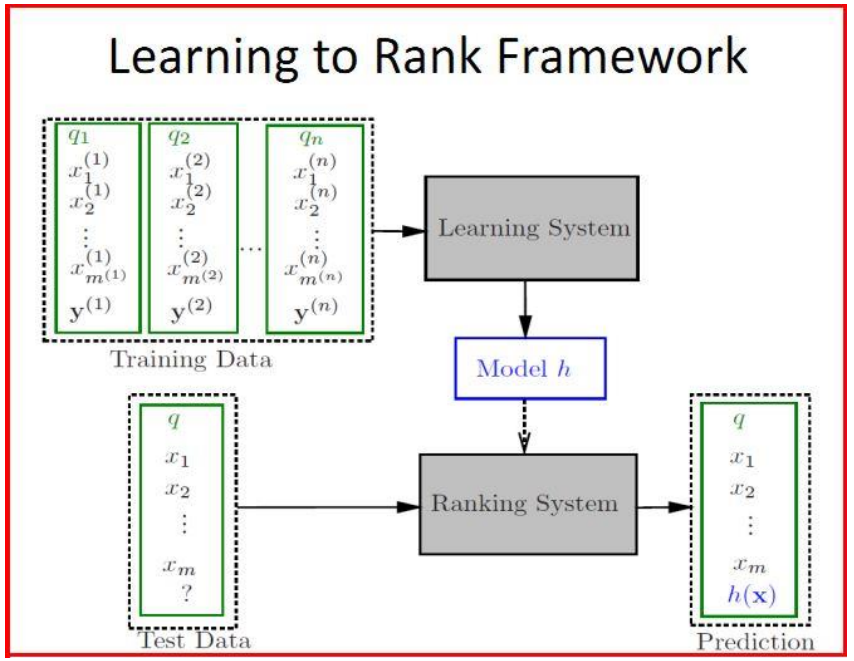
- 相比于PageRank，HITS算法是一种能够区分网页功能的排序算法
- 优点
  - 更好地描述互联网组合特点
  - 主题相关，因此可以单独用于网页排序
- 缺点
  - 需要在线计算，时间代价较大
  - 对链接结构变化敏感，且依然可能受到“链接作弊”的影响

本节目录

- 信息检索与相关度计算
- 网页权威性计算
- **排序学习问题（引子）**

- **从网页排序到排序学习**

- 对于网页内容的排序问题，本质上是一个学习排序器（Ranker）的问题
  - 输入网页（文档）内容及查询，输出该网页合适的排序位置
  - 排序学习是个有监督学习问题
    - 基于已知的排序（如用户反馈）
    - 为新网页-查询对给出排序



- **从网页排序到排序学习**

- 对于网页排序而言，排序学习问题有着独有的特性
  - 最重要的一点：查询决定了排序学习的逻辑结构
    - 排序的损失函数应设计在查询层面，不同函数得到不同Ranker
  - 位置更为敏感，排在前部（尤其顶端）的部分更重要
    - 用户可能只看开头的几个页面
  - 评价指标基于排序结果进行，不是简单的分类问题

- **从网页排序到排序学习**

- 三类常见的排序学习算法
  - Pointwise，将排序退化为分类或回归问题
    - 输出：网页对应的分类（有序）、回归值或有序回归值
  - Pairwise：比较一对网页之间的相关度，e.g., 相关 > 不相关
    - 输出：网页对之间的偏序关系
  - Listwise：对整个网页集合进行排序
    - 输出：整个集合的完整排序，往往依赖特定排序指标

本章小结

网页排序

- 信息检索模型概述与相关度计算
  - 基于集合论/代数论/概率论的不同计算模型
- 网页权威性评估
  - PageRank及其拓展算法：个性化、主题敏感、Hilltop
  - HITS算法与网页角色分工
- 排序学习问题（Learning to Rank）

tongxu@ustc.edu.cn