第 38 卷 第 2 期
Vol.38 No.2

计 算 机 工 程
Computer Engineering

2012 年 1 月
January 2012

• 软件技术与数据库 •　　文章编号：1000—3428(2012)02—0036—03　　文献标识码：A　　中图分类号：TP301.6

一种基于聚类的语义检索算法

向河林，张明西，李珀瀚，何震瀛，汪 卫

(复旦大学计算机科学技术学院，上海 201203)

摘 要：潜在语义分析在进行大规模语义检索时计算效率较低、存储开销较大。针对该问题，提出一种基于聚类的潜在语义检索算法。通过文档之间的结构关系对文档进行聚类，利用簇代替文档分析潜在语义，以此减少处理文档的个数。实验结果表明，该算法能减少查询时间，且检索精确度较高。

关键词：潜在语义分析；信息检索；向量空间模型；图聚类算法

Clustering-based Semantic Retrieval Algorithm

XIANG He-lin, ZHANG Ming-xi, LI Po-han, HE Zhen-ying, WANG Wei

(School of Computer Science, Fudan University, Shanghai 201203, China)

【Abstract】Latent Semantic Analysis(LSA) lacks computation efficiency and has storage deficiencies when it is used in the large scale semantic retrieval. To solve this problem, this paper proposes a clustering-based semantic retrieval algorithm. This algorithm clusters the documents using their structural information, and applies the LSA process on those clusters to efficiently reduce the number of documents. Experimental results show that the algorithm can exponentially decrease the time of inquiring and get good retrieval accuracy.

【Key words】Latent Semantic Analysis(LSA); information retrieval; vector space model; graph clustering algorithm

DOI: 10.3969/j.issn.1000-3428.2012.02.011

1 概述

信息检索技术在网络时代得到了广泛的应用和发展。潜在语义分析(Latent Semantic Analysis, LSA)是一套通过统计分析提取文档中词语之间潜在语义信息的理论和方法[1-2]。该模型通过挖掘出词语与词语之间、词语与文档之间的语义层面关系，提供给用户在语义层面上的检索，以解决信息检索中存在的一义多词的问题。虽然 LSA 在语义层面的检索上有着十分理想的效果，但是在实际应用中却存在一些局限，即在大规模数据集下，LSA 存储矩阵的空间开销较高，同时检索的时间效率也较低；现有的 LSA 模型是基于文本内容的语义分析，没有考虑文档和文档之间的结构关系。基于以上内容，本文提出一种基于聚类的 LSA 算法(Clus-LSA)。

2 相关工作

潜在语义分析是 1990 年由文献[1]提出的一种新的索引和检索方法。最初用于解决关键字检索中的同义词和多义词问题。潜在语义分析的基本思想认为词与词之间存在的联系即语义结构隐含在文档中，因而采用统计计算的方法，通过对大量的文本进行分析找出这种潜在的语义结构。进行潜在语义分析过程一般较为流行的方法是，采用奇异值分解(Singular Value Decomposition, SVD)技术处理词-文档向量空间矩阵。文献[2]将 LSA 的思想带入到生成概率的统计模型中，提出 PLSA(Probalistic LSA)模型，其不同于 SVD 的追求矩阵间方差和最小的原则，而是基于最大可能性原则，具有更坚实的统计学基础。潜在语义分析被广泛应用到各领域，如文献[3]将潜在语义分析技术应用到个性化的查询扩展中。

聚类分析也称非监督的分类方法，旨在发现数据中所描述的对象之间的关系，尽可能将相似的对象进行归类，将不相似的对象尽可能地分开。图聚类是一类特殊的聚类，其主要的特殊性在于，在度量节点的相似度上，其使用更多的是图的拓扑结构性质，而非对象本身的属性。在图节点相似度度量的问题上，文献[4]提出 SimRank 算法。该算法是一种基于结构信息度量节点相似度的方法，其基本思想是“每 2 个节点的相似度由其链接节点所对应相似度的平均值决定”，并且提出一种通过不断地运用公式迭代计算图中节点间相似度方法。文献[5]继承 SimRank 的思想，提出一种基于层次树 SimTree 用来聚类不同类节点的方法。该方法利用层次树的结构，以合并大量节点间相似度的存储和计算，在牺牲一定精度的同时换来较大范围的性能提升。

3 传统 LSA/SVD 过程

假设在一个信息检索系统的文档集合 D 中，包含 n 个文档 $\{d_1,d_2,\cdots,d_n\}$；词语集合 T 中包含 m 个词 $\{t_1,t_2,\cdots,t_m\}$；现在给出查询关键词的集合 Q 包含 p 个关键词 $\{q_1,q_2,\cdots,q_p\}$，要求返回 D 中与 Q 最相似的前 K 个结果并排序。

LSA 沿用了传统的向量空间模型，利用 SVD 分解技术压缩向量空间。具体过程如下：

(1)文档分析，词-文档矩阵 $\boldsymbol{C}$ 构造

将文档 d_i 表示成向量的形式：$\boldsymbol{V}_i=(v_1,v_2,\cdots,v_m)$，$v_j$ 为 t_j 在 d_i 的文档中的权重(可以为简单的词频统计值，也可以是 TF-IDF 值)，于是词-文档矩阵 $\boldsymbol{C}$ 可以表示为：$\boldsymbol{C}=(\boldsymbol{V}_1^{\mathrm{T}},\boldsymbol{V}_2^{\mathrm{T}},\cdots,\boldsymbol{V}_n^{\mathrm{T}})$。

(2)SVD 分解运算

设 $\boldsymbol{C}$ 是 $m\times n$ 大小的矩阵，秩为 r；$\boldsymbol{U}$ 为 $m\times n$ 大小的矩

基金项目：国家自然科学基金资助项目(60703093)

作者简介：向河林(1986－)，男，硕士，主研方向：数据挖掘，信息检索；张明西，博士；李珀瀚，硕士；何震瀛，讲师、博士；汪 卫，教授、博士生导师

收稿日期：2011-07-22　**E-mail：**082024016@fudan.edu.cn

阵，其中，$\boldsymbol{U}$ 的列为矩阵 $\boldsymbol{CC}^{\mathrm{T}}$ 的正交特征向量；$\boldsymbol{V}$ 为 $n\times n$ 大小的矩阵；$\boldsymbol{V}$ 的列为矩阵 $\boldsymbol{C}^{\mathrm{T}}\boldsymbol{C}$ 的正交特征向量。存在奇异值分解：$\boldsymbol{C}=\boldsymbol{USV}^{\mathrm{T}}$。其中，$\boldsymbol{S}$ 为 $m\times n$ 矩阵，$\boldsymbol{S}_{ii}=\sqrt{\lambda_i}$；$\lambda_i$ 为矩阵 $\boldsymbol{CC}^{\mathrm{T}}$ 的第 i 大的特征值。

(3)$\boldsymbol{C}$ 的低阶近似矩阵求解

取 $k<<r$，计算：$\boldsymbol{C}_k=\boldsymbol{US}_k\boldsymbol{V}^{\mathrm{T}}$ 作为 $\boldsymbol{C}$ 的 k 阶近似矩阵，其中，$\boldsymbol{S}_k$ 为将 $\boldsymbol{S}$ 的后 $r-k$ 行置为 0 得到的矩阵。$\boldsymbol{C}_k$ 的 n 个列向量为 n 个文档在新的 k 维空间下的向量表示。由于越小的特征值对矩阵的影响越小，因此文献[1]证明这样取得的 $\boldsymbol{C}_k$ 满足 F-范数最小。

(4)在新的 k 维空间下进行语义检索

求出关键词向量 $\boldsymbol{Q}$ 对应的低阶向量坐标：$\boldsymbol{Q}'=\boldsymbol{V}^{\mathrm{T}}\boldsymbol{S}_k^{-1}\boldsymbol{Q}$。余弦相似度计算式如下：

$$Sim(\boldsymbol{Q}',d)=\frac{\sum w_{\boldsymbol{Q}',i}\times w_{\boldsymbol{d},i}}{\sqrt{\sum w_{\boldsymbol{Q}',i}}\times\sqrt{\sum w_{\boldsymbol{d},i}}} \tag{1}$$

其中，$w_{\boldsymbol{Q}',i}$ 表示向量 $\boldsymbol{Q}'$ 的第 i 个分量；$w_{\boldsymbol{d},i}$ 则表示向量 $\boldsymbol{d}$ 的第 i 个分量。

利用式(1)求出 $\boldsymbol{Q}'$ 与 $\boldsymbol{C}_k$ 的 n 个列向量的相似度，根据相似度大小进行排序得到检索的最后结果。

4 算法设计

传统潜在语义分析模型在应用到信息检索系统的时候存在两方面局限性，即时间和空间的效率表现较差；没有将文档之间的结构相似性考虑进来。本文的解决思路是通过聚类的方法对在结构上存在较高相似性的文档合并，再对合并后的簇信息进行 SVD。这样做能成倍地减少查询的时间和空间开销，大幅度提升查询的性能。

4.1 基于结构特性的图聚类算法

从图的角度去理解潜在语义分析模型，将文档和词看成 2 类节点，则整个词-文档向量空间可以被看成是一个二分图(Bipartite Graph)，因为只存在文档节点和词节点之间有边。但是在实际情况中，在文档节点之间可能存在着某种关联，例如文档之间的引用信息等。将这类信息看成基于结构的信息，并且将文档和词之间的边看成是基于内容的。根据这些基于结构的信息对文档进行聚类，以减少文档的个数，以此避免重复计算。

基于结构特性的聚类算法需要有基于结构特性的相似度度量规则。在一般的聚类算法中，通常用几何距离，汉明距离或者余弦相似度方法等来度量 2 个实体之间的相似度。但在有向图的聚类中，度量 2 个节点的相似度不能使用通常的直观方法。文献[4]中给出一种通过图的结构信息来度量节点之间相似度的方法。其基本思想是：如果连接到 2 个节点(或者被 2 个节点连接)的节点是相似的，那么这 2 个节点也是相似的。

定义 1 假设 a 和 b 为有向图中的 2 个点，a 和 b 的入链相似度为：

$$SI(a,b)=\frac{C_1}{|I(a)||I(b)|}\sum_{i=1}^{|I(a)|}\sum_{j=1}^{|I(b)|}SI(I_i(a),I_j(b)) \tag{2}$$

a 和 b 的出链相似度为：

$$SO(a,b)=\frac{C_2}{|O(a)||O(b)|}\sum_{i=1}^{|O(a)|}\sum_{j=1}^{|O(b)|}SO(O_i(a),O_j(b)) \tag{3}$$

其中，$I_i(a)$ 为节点 a 的第 i 个入连接节点；$O_i(a)$ 为节点 a 的第 i 个出连接节点；C_1 和 C_2 是 0~1 之间的常数。需要说明的一点是，如果 $|I(a)|$ 或者 $|I(b)|$ 为 0，亦即如果 a、b 中任意一个节点的入连接节点为空，则有 $SI(a,b)=0$；同理，若 $|O(a)|$ 或者 $|O(b)|$ 为 0，则有 $SO(a,b)=0$，上述即为入链相似度和出链相似度的定义。

定义 2 假设 a 和 b 为有向图中的 2 个点，则 a 和 b 的相似度为：

$$S(a,b)=p\times SI(a,b)+(1-p)\times SO(a,b) \tag{4}$$

其中，p 是 0~1 之间的常数，以上为节点间相似度的定义。

根据以上的定义，以计算入链相似度为例，给出一个迭代计算相似度的步骤，假设有如下计算式：

(1)初始化相似度

$$SI_0(a,b)=\begin{cases}0 & \text{if} \quad a\neq b\\ 1 & \text{else}\end{cases} \tag{5}$$

(2)相似度迭代

$$SI_{k+1}(a,b)=\frac{C_1}{|I(a)||I(b)|}\sum_{i=1}^{|I(a)|}\sum_{j=1}^{|I(b)|}SI_k(I_i(a),I_j(b)) \tag{6}$$

在初始化阶段生成距离不超过半径的节点对，并且按照式(5)初始化相似度。根据与之相邻的节点对的相似度更新当前节点对的相似度。C_1 和 C_2 是一个常数，通常都取 0.8，迭代次数一般取 5 次~6 次。文献[4]从理论和实验上都证明大约 5 次左右的迭代后，相似度能得到较为精确的值。半径 R 为节点对之间的最长的最短路径长度，通常取 2 或 3。假设有文档节点及引用结构信息，迭代次数 k，半径 R，则计算相似度的迭代算法步骤如下：

(1)遍历节点，生成所有距离不超过 $2\times R$ 的节点对，并按照式(5)初始化相似度。

(2)i 从 1 取到 k，遍历生成的节点对，按照式(6)更新节点对的入链相似度和出链相似度。

(3)遍历每对节点对，进行步骤(4)。

(4)按照式(4)计算出节点对的相似度。

基于以上算法，根据 N 个文档节点，节点对的相似度，每个簇包含的文档个数上限 L，相似度阈值 ε，提出凝聚的层次聚类算法步骤如下：

(1)初始化 N 个簇，每个簇中包含一个原始的文档节点。

(2)将 N' 赋初始值 N/L。

(3)i 从 1 取到 N'，依次进行步骤(4)。

(4)遍历找到 2 个相似度最大的簇 A, B，即包含的节点个数之和最少且小于等于 L，进入步骤(5)。

(5)如果找到了符合条件的 A 和 B，且 $S(A,B)\geqslant\varepsilon$。将 A 和 B 合并，更新簇(A, B)和其他簇之间的相似度信息。

(6)否则终止步骤(3)~步骤(5)，进入步骤(7)。

(7)i 从 1 取到 $|C|$，依次进行步骤(8)。

(8)对簇 C_i 内的文档进行相似度排序。

凝聚的层次聚类算法的基本思想是：将每个点看成是一个簇。不断地合并最近的簇，直到所有对象都在一个簇里或者某个终止条件被满足。

在聚类的过程中会产生簇，也就是同类点的集合。簇间的相似度采用全连接的方法度量，即 2 个簇之间的相似度由其包含的每对节点相似度的平均值来表示。

定义 3 (簇间相似度)设有 2 个簇 A, B。其中，A 包含 $a_1,a_2,\cdots,a_s$ 一共 s 个点；B 包含 $b_1,b_2,\cdots,b_t$ 一共 t 个点。则 A 和 B 的相似度定义为：

$$S(A,B)=\frac{1}{s\times t}\sum_{i=1}^{s}\sum_{j=1}^{t}(a_i,b_j) \tag{7}$$

在聚类的初始化阶段，由于每个簇只包含一个节点，因此簇间相似度即为节点间的相似度(凝聚的层次聚类算法步骤(1))。每次迭代过程找到 2 个相似度最大的簇，对其进行合并(凝聚的层次聚类算法步骤(3)、步骤(4))。设置 2 个参数 L 和 ε: L 为每个簇包含的文档个数上限；为避免每个簇的节点个数过多，ε 为相似度阈值，相似度低于 ε 的 2 个簇不进行合并(凝聚的层次聚类算法步骤(5))。凝聚的层次聚类算法步骤(6)说明如果在迭代的过程中找到的相似度最大的 2 个簇其相似度仍然小于 ε，则停止聚类过程。

4.2 Clus-LSA 算法框架

结合聚类的 Clus-LSA 算法框架如下：

(1)将文档进行聚类。按照 4.1 节基于结构特性的图聚类算法合并相似文档，得到 K 个簇，其中每个簇中包含不超过 L 个文档，并且其内部是按照相似度大小排序。

(2)构建词-簇矩阵 $\boldsymbol{M}$。$\boldsymbol{M}$ 中的列向量为簇向量，每个簇向量表示为其包含的文档向量之和，即：

$$\boldsymbol{V}_c = \sum_{d_k \text{ in } C} \boldsymbol{V}_{d_k}$$

(3)对 $\boldsymbol{M}$ 进行传统的 LSA/SVD 过程。得到基于聚类信息的 SVD 分解矩阵：

$$\boldsymbol{C}_k = \boldsymbol{U}\boldsymbol{S}_k\boldsymbol{V}^{\mathrm{T}}$$

基于簇的关键词检索过程类似于传统的关键词检索。需要说明的是在检索前，将关键词伪文本向量 $\boldsymbol{Q}'$ 与簇聚类得到的 SVD 矩阵进行余弦相似度比较，以得到按照余弦相似度排序后的簇序列，然后将簇内部的文档按照聚类过程中排好的序列代替簇展开即得到近似的文档排序。

传统 LSA 检索的时间复杂度理论上为 $O(N)$，N 为文档的个数。进行聚类后文档的个数变为 K，其中，L 为聚类中每个簇能容纳的文档个数的最大值。虽然理论上的时间复杂度依然是 $O(N)$，但是从常数看将原来的时间减少为原来的 K/N，在实际检索中，这样程度的性能提升是十分有意义的。

5 实验结果与分析

在本文实验中，使用的机器的 CPU 为 Intel Core Duo2 E7500 2.93 GHz，内存为 2 GB。实验使用的操作系统包括 Windows XP 以及 64 位 Ubuntu 10.04 版。其中，聚类算法及检索算法在 XP 下用 C++实现，SVD 算法在 Ubuntu 64 位下用 C++实现。

对于 SVD 算法选取参数 K=400。而对于聚类算法选取参数 L=10，ε=0.06。

实验数据集是 SNAP(http://snap.stanford.edu/data/#citnets)上的 Citation 数据集，包含 27 770 篇在 arxiv(http://arxiv.org)上发表的从 1997 年－2003 年期间的高能理论物理领域里的会议文献。基于聚类 LSA 算法(Clus-LSA 算法)和传统 LSA 算法，在检索关键词“quantum”(量子)中，时间性能的比较结果如图 1 所示。

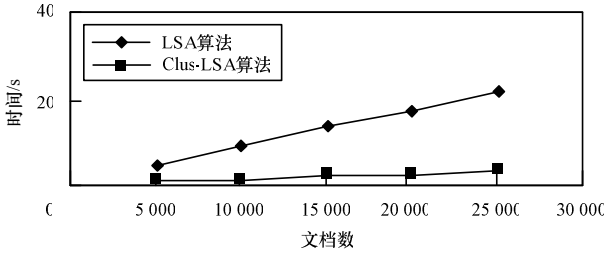

图 1 2 种算法在检索关键词“quantum”上的时间性能对比

由图 1 可知，查询响应时间与文档数基本成线性关系，这符合关于检索的时间复杂度为 $O(N)$的理论分析的结论。由于 Clus-LSA 算法将文档数成倍地减少，因此成倍减少了查询时间。此外，图 1 中 2 条线的斜率可以反映出 Clus-LSA 算法的时间开销随文档数的增长比传统 LSA 算法的增长要缓慢很多。

选取 4 组关键词，分别为 big bang、gravitation、quantum mechanics、string theory，取检索结果的前 10 名与关键词进行相关个数(指关键词与检索结果中的文档集合的相关个数)对比，2 种算法相关个数对比如图 2 所示。可知 Clus-LSA 算法检索的相关个数与 LSA 算法相差不大，这说明 Clus-LSA 算法的检索结果在精度上存在一定损失，但仍然在可接受范围内。

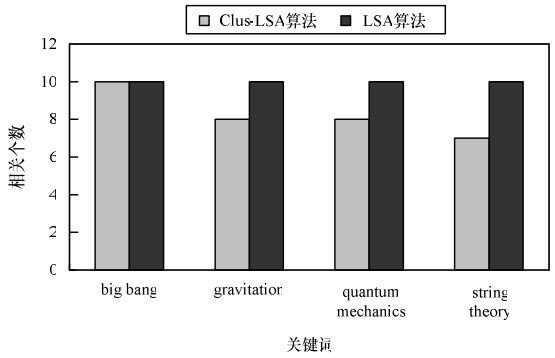

图 2 2 种算法相关个数对比

6 结束语

本文设计一种基于聚类的语义检索算法，以解决传统潜在语义分析算法在检索效率等方面的不足。

由于加入文档之间的结构信息，在实验中发现其检索精准度会有所下降，下一步的研究重点为，如何将簇内文档的相似度和 LSA/SVD 计算出来的簇相似度相结合，提出一个完整的排序模型，使结果的精确度得到进一步提升。

参考文献

[1] Deerwester S, Dumais S T, Furnas G W, et al. Indexing by Latent Semantic Analysis[J]. Journal of the American Society for Information Science, 1990, 41(6): 391-407.

[2] Hofmann T. Probabilistic Latent Semantic Indexing[C]//Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New York, USA: ACM Press, 1999.

[3] 王卫国, 徐炜民. 基于潜在语义分析的个性化查询扩展模型[J]. 计算机工程, 2010, 36(21): 43-45.

[4] Jeh G, Widom J. SimRank: A Measure of Structural-context Similarity[C]//Proc. of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA: ACM Press, 2002.

[5] Yin Xiaoxin, Han Jiawei, Philip Y S. LinkClus: Efficient Clustering Via Heterogeneous Semantic Links[C]//Proc. of the 32nd International Conference on Very Large Data Bases. Seoul, Korea: [s. n.], 2006.

编辑 刘 冰