基于划分的 XML 文档聚类研究*)

杨厚群[1,2]　何中市[1]　雷景生[2]

(重庆大学计算机学院　重庆 400044)[1]　(海南大学信息科学技术学院　海口 570228)[2]

摘　要　本文在文本聚类的基础上对 XML 文档聚类进行了研究，对划分聚类法进行了改进，使之适合于 XML 文档聚类。最后通过路径划分聚类算法根据频繁结构对 XML 文档进行挖掘聚类，并对实验结果进行讨论。

关键词　划分聚类，数据挖掘，XML

Research of Clustering XML Documents Based on Partition

YANG Hou-Qun[1,2]　HE Zhong-Shi[1]　LEI Jing-Sheng[2]

(College of Computer Science, Chongqing University, Chongqing 400044)[1]

(College of Information Science & Technology, Hainan University, Haikou 570228)[2]

Abstract　On the basis of text cluster, this paper makes an exploratory research on XML documents clustering, through the improvement on partition clustering based on the XML structural representation by XPath, makes them can use on XML documents cluster. Finally, this paper discusses testing results.

Keywords　Clustering, Data mining, XML

1　引言

聚类方法被广泛研究并应用于机器学习、统计分析、模式识别以及数据挖掘与知识发现等不同的领域，是分析数据并从中发现有用信息的一种有效手段。随着 Internet 的发展，它所蕴含的海量信息大大超过了用户可能的阅读量。这些信息很多是以 XML 作为数据表示和交换的标准。组织这些数据并从中获取所需信息，能促进对 XML 文档聚类的研究。由于 XML 的结构蕴含语义，一些研究者建议[1,6]利用其对 XML 文档进行聚类，XML 文档聚类可广泛应用于文本挖掘与信息检索的不同方面。

2　背景

2.1　聚类问题

聚类能够在没有训练样本的条件下自动产生聚类模型。作为数据挖掘的一种重要手段，聚类在 XML 文档的信息挖掘中起着非常重要的作用，它不但可以提高信息检索系统的查准率和查全率，还可以用于组织搜索引擎返回的结果，自动产生文档的层次簇或类，并利用这些簇或类来对新文档进行归类。

目前文档聚类的方法主要有划分聚类法、层次聚类法、自组织映射法、基于遗传算法的文档聚类法等。XML 文档聚类的目标和普通的文档聚类一样，就是将 XML 文档集合分成若干个簇，要求簇内文档内容的相似性尽可能大，而簇之间文档的相似性尽可能小。由于 XML 文档是一种结构化的文档，其语义信息可以通过文档结构加以描述，主要是通过对 XML 文档结构的分析比较进行聚类。本文采用的划分聚类方法在文档聚类中较为常用，是一种主要的文档聚类方法。

2.2　XPath

对 XML 文档内部对象的定位是一个重要的部分。W3C 推出的 XPath[2]正是这样一种定位语言，它建立在 XML 文档树的抽象层次之上，为各种节点对象位置的描述提供语法和相关语义。XPath 的定位机制与传统的文档或数据库管理系统中的定位机制是有所区别的，区别的实质在于 XPath 具有更加严格的面向文档层次结构的特性，基于这样的特性，XPath 提供了许多面向文档实例的模式定位手段。XML 文档可以看作是点标记的有向树。XPath 作为 XML 树中导航查询的基本机制，支持丰富的路径查询特性。本文讨论的 XPath 基于下列语法定义：

$P::=/E \mid //E$

$E::=label \mid text() \mid * \mid @* \mid E/E \mid E//E \mid E[Q]$

$Q::=E \mid E\ Operl\ Const \mid Q \text{ and } Q$

$Oprel::=\langle \mid \pounds \mid \rangle \mid {}^{3} \mid = \mid {}^{1}$

非形式化地，本文利用 XPath 中出现的操作符号的语义。给定 XML 数据树，“/”表示数据节点之间的父子关系，“//”表示节点之间的祖孙关系，“[]”表示路径之间的条件关系，“@”表示 XML 元素的属性，“*”表示任意的数据元素。另外，支持在路径表达式中定义逻辑表达式，包括“$\langle \mid \pounds \mid \rangle \mid {}^{3} \mid = \mid {}^{1}$”。

一些研究者在 XPath 的应用上做了许多工作。Georg Gottlob 分析了目前 XPath 执行器的执行流程[3]，提出了一种优化的自底向上的物理执行策略，提高了 XPath 的执行效率。S. Amer-Yahia 提出了简化 XPath 的方法[4]，主要完成了将包含大量节点的查询转换为包含少量节点的等价查询。Frank Neven[5]给出了支持不同特性的 XPath 在 DTD 下包含判定的复杂性。Michael Benedikt[7]分析了支持不同特性的 XPath 的查询表达能力，同时基于树模式和 XPath 的逻辑语义，给出了 XPath 在常规操作，如交、并、取反操作下的闭包性。本文借鉴了其中的一些方法。

*)海南省教育厅高校科研项目(Hjkj200603)。杨厚群　副教授，博士生，主要研究方向为数据挖掘与机器学习。

3 XML 文档聚类

XML 文档可以基于挖掘出的频繁结构来进行描述。每个 XML 文档可表示为一个带有根和标签的树，只考虑 XML 文档结构，不考虑其元素的值。树的每个节点都与标签相关联，用于表示 XML 文档中的一个元素。树结构可用几种方式来表现，当然也可以基于路径表现。在基于路径的表现中，一个树就是一个路径集，每个树由一个根及其子元素(即内部子元素和叶节点)构成。一个 XML 文档由一个路径集合表示，因此多个 XML 文档形成了一个大的路径集合。本文用一个简单 XPath 表达一条路径。XML 文档可用一个从根到叶节点的简单 XPath 表达式集合来表现。这里面不仅包含路径元素的次序信息，而且包含路径中元素所属层次的信息。通过挖掘算法进行挖掘，如果发现路径是频繁的，那么 XML 文档的结构相似度将取决于相似路径的数量及路径所在层级的计算结果。最终，相似的文件将被聚集在同一组。

可将基于简单 XPath 的路径表达式视为序列，而利用序列模式挖掘算法[8]可以用于找到频繁序列(树结构)。每个序列对应于树层一个降序排列的元素集合，并由一个序列 id 标记。一个 XML 序列如果是另一个序列的子序列，那么它就能被另一个序列所包含。在 XML 序列集中，一个 XML 序列 s_j 如果没有被其他任何序列所包含，那么它就是最大的序列。

3.1 路径挖掘算法

给定 XML 序列的数据库 D，其中包括下列各域：文档标记(doc-id)、序列标记(path-id)、元素(tag)和相应级别的树层(t-level)。doc-id 和 path-id 组合形成一个 XML 序列的标识符，t-level 则是元素(tag)在其相应序列里的位置。根据挖掘算法[9]，设 s_j 是 D 中的一个 XML 序列，$O(s_j)$ 是 s_j 的出现次数。本文并没有把在一个 XML 文档中找到的相同序列的数量考虑进去。在 D 中 s_j 频繁度(支持度)定义为

$$freq(s_j)=O(s_j)/m$$

m 是 D 中文档的数量。设 $0\leqslant\delta\leqslant1$，如果 $freq(s_j)\geqslant\delta$，那么 XML 序列 s_j 就是频繁的。根据支持度$\geqslant\delta$ 的要求，在 XML 序列中找出频繁的 XPath，这些频繁的 XPath 并不包含在另外的用于表达 XML 文档的频繁结构的 XPath 中。最小支持度 δ 定义了频繁结构的解决方案，它的输入是一个简单 Xpaths 表达式(XML 序列)，而输出则是一个最大的频繁路径集合。具体的挖掘算法可以在文[9]找到。

在 D 中使用基于 XPath 挖掘算法挖掘频繁路径，然后用 XPath 描述文档的特征。n 为使用最小支持度 δ 挖掘出的 XPath 的数量。采用合适的距离函数，如欧氏或曼哈顿距离计算 XML 文档之间的相似度。在具体实现中，本文运用了划分聚类[10]对 XML 文档进行聚类。划分聚类法通过一个评估函数将文档树集水平地分割为若干个类。具体算法描述如下：

输入：XML 文档集合 $D=\{d_1,\cdots,d_m\}$，minsup：$0\leqslant\delta\leqslant1$
输出：聚类树
步骤 1：通过简单 XPath 集合描述一个 XML 文档，并设定 minsup 的大小。
步骤 2：找出第一个频繁元素集 L_1，从 L_1 中将每个路径转换为用图表表示的形式(不满足最小支持度的被删除)。
步骤 3：找出所有频繁 XPath 表达式集合，并找出其中最大的 XPath，构建树(XML 文档 D)。
步骤 4：确定要生成的类的数目 k，按照某种原则(可随机)生成 k 个聚类中心作为聚类的初始中心点 $S=\{s_1,s_2,\cdots,s_n\}$。
步骤 5：对文档集中的每个文档 d_1，依次计算它与各个中心点 s_j 的相似度 $sim(d_i,s_j)$；并选取具有最大的相似度的中心 $argmaxsim(d_i,s_j)$ 点，将 d_1 归入以其为聚类中心的类 C_j，从而得到 D 的一个聚类 $C=\{C_1,\cdots,C_k\}$。
步骤 6：重新计算每个类的中心点。
步骤 7：重复步骤 5，6，直至类中心点不再改变，得到稳定的聚类结果。

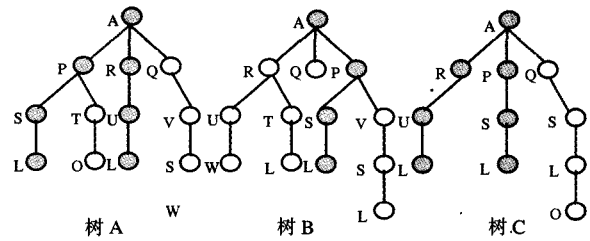


图 1 Xpath for minsup=0.5

计算 D 中 d_1 与中心点的相似度实质上是计算两个文档之间的相似度。$C=\{C_1,\cdots,C_k\}$ 是文档 D 的一个聚类，要找出其中心点，可以通过计算文档间距离的平方和来完成，计算公式如下[14]：

$$S(d_i,C)=\sum_{k=1}^{n}O^2(d_i,d_k)$$

$O(d_i,d_k)$ 是 d_i 和 d_k 的距离。对 C 中所有文档进行计算后，挑选出 S 最小的那个文档 $\mathrm{Min}(S(d_i,C))$，作为新的聚类中心点。

该过程构造出一个树状结构，其中包含了类的层次信息以及所有类内和类间的相似度。划分聚类法的运行速度较快，但必须事先确定 k 的取值和初始中心点，而且初始中心点的选择对最终的聚类结果有较大影响。

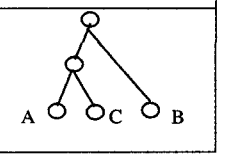
Doc	Path1 A/P/S/L	Path2 A/R/U/L	Path3 A//O	Output
A	1	1	1	
B	1	0	0	
C	1	1	1	

图 2 输出 XML 树

类对之间相似度的计算可以分为两步：首先计算单个文档和类之间的相似度，再计算总的相似度。假设 (C_i,C_j) 为一个类对，$C_i=\{d_i,\cdots,d_m\}$，$C_j=\{d_i,\cdots,d_n\}$，$S=\{d_i,\cdots,d_j\}$ 为文档 d_i 和 d_j 之间的相似度，则有

$$Sim(C_i,C_j)=\sum_{i=1}^{m}\sum_{k=1}^{n}S^2(d_i,d_k)$$

3.2 聚类结果的应用

XML 文档的聚类结果可以用来解决操纵 XML 文档的各种实际问题，例如索引和集成[11]。基于上面的聚合聚类，一个 XML 文档的聚类 C_d 可以表达为路径的集合，它们表达了整个文档域的语义和知识簇，这样的基于知识的聚类模式可以用来帮助检索读者的兴趣话题。比如通过一个路径，读者要从数字图书馆检索有关文件(包括相关的问题)。这个解决方案可以通过聚类表达如下。

(1)获得用户的特定主题(路径)的聚类；
(2)通过聚类簇计算所得到的支持度；
(3)通过其最高支持度识别该簇；
(4)返回相关主题和文档的聚类。

对于 XML 文件的索引，现有的逆序检索对路径短的 XML 查询效率较高，但对路径长的 XML 检索效率不高，这是由于响应时间对路径长度来说是指数级的。如果用聚类算

法对 XML 文件分组，就可以为每个聚类创造一种倒排文件。这个倒排文件可将不同片段进一步划分为频繁路径及树的代表，整个树对应一条路径的后缀，路径上各节点的子节点和父节点合并。这能够加快 XML 数据库文件的搜索。

4 实验结果分析

实验数据来自于 SigmodXML 数据集[12]，在实验中，已知 XML 数据集中的文档分属于 4 类(根据 DTD 确定)，每一类中的文档数目也已计算出来。并和 Tag-based 算法[6]、TreeFinder 算法[13]进行了比较。本文采用了通用的查准率和查全率来评价这些聚类结果，并引入一个内部度量标准来考查已发现的聚类簇的紧凑性，即

$$IC(P)=\frac{1}{n}\sum_{C_i\in P}\frac{1}{n}\sum_{d\in C_i}Dist(d,rep(C_i))$$

式中，$P=\{C_1,\cdots,C_i,\cdots,C_n\}$，$P\in D$，$C_i=\{d_i,\cdots,d_m\}$，$rep(C_i)$是$(C_i,C_j)$的聚类代表。

表 1 实验结果

文档	平均大小	DTD	算法	聚类	Precision	Recall	IC
84	3.2k	4	Tag-based	10	0.873	0.983	0.310
			TreeFinder	10	0.965	0.976	0.215
			XPath	10	0.974	0.983	0.331
84	2.9k	4	Tag-based	10	0.804	0.907	0.345
			TreeFinder	10	0.854	0.869	0.298
			XPath	10	0.865	0.968	0.318
84	4.3k	4	Tag-based	10	0.680	0.916	0.384
			TreeFinder	10	0.852	0.785	0.343
			XPath	10	0.856	0.922	0.391

三个文档集组成了测试数据，每个包含 84 个文档(每个类别 21 个文档)。表 1 就表明了该算法区分不同大小文档的准确性。可以看出，Tag-based 算法限制比较宽，聚类的内聚度很低，对较大的 XML 文档效果较好；TreeFinder 算法要求比较严格，聚类的内聚度比较高，但是文档的查全率比较低，对于较小的 XML 文档效果很好。基于 XPath 的算法，则显示出了良好的适应性。

结束语 基于路径的 XML 文档聚类是当前文本挖掘研究中的一个全新领域，由于其处理对象是结构化的 XML 文档，所以具体的聚类方法和一般的文本聚类有着较大差别。特别是 XML 文档的语义信息可借助于 XPath 描述的文档结构得以表示，通过增加文档间相似度比较的准确度和精确度，可以更加便利地操纵 XML 文档为用户的查询和检索提供服务。如何利用聚类得到的路径信息来提高 XML 信息检索效率，是今后继续研究的重点。

参 考 文 献

1 Cheng J, Yu G, Yu J X, et al. An EfficientClustering Based Indexing Method for XML PathExpressions. In: Proc. 8th Int Conf on Database Systems forAdvanced Applications, Kyoto, March, 2003
2 Clark J, DeRose S. XML Path Language (XPath), version 1. 0, 1999. http://www. w3. org/TR/1999/ REC- xpath-19991116
3 Georg G, Christoph K, Reinhard P. Efficient algorithms for processing XPath queries. In: Stéphane B, Akmal B. eds. Proc. of the VLDB 2002. Heidelberg: Springer-Verlag, 2002. 95～106
4 Sihem A, SungRan C, Laks V S. Minimization of tree pattern queries. In: Walid G A, eds. Proc. of the SIGMOD. Santa Barbara, 2001. http://www. research. att. com/～ sihem/publications/SIGMOD01. pdf
5 Frank N, Thomas S. XPath containment in the presence of disjunction, DTDs, and variables. In: Diego C, Maurizio L, eds. Proc. of the ICDE. Heidelberg: Springer-Verlag, 2003. 315～329
6 Guillaume D, Murtagh F. Clustering of XMLdocuments. Computer Physics Cmmunications, 2000, 127(2-3): 215～227
7 Gerome M, Dan S. Containment and equivalence for an XPath fragment. In: Lucian P, ed. Proc. of the PODS. ACM, 2002. 65～76
8 Agrawal R, Srikant R, Mining Sequential Patterns. In: Proceedings of the Eleventh International Conference on Data Engineering, Taipei, Taiwan, March 1995. 3～14
9 Leung H P, Chung F L, Chan S C F. On the use of hierarchical information in sequential mining-based XML document similarity computation. Knowledge and Information Systems, 2005, 7(4)
10 Kaufman L, Rousseeuw P J. Finding Groups in Data: An Introduction to Cluster Analysis. New York: Wiley, 1990
11 Lee M L, et al. XClust: Clustering XML schemas for Effective Integration. In: Proc. 11th Int Conf on Information & Knowledge Management, McLean, Nov. 2002. 292～299
12 Sigmod XML DataSet. Available at : http :/ / www. acm. org/ sigmod/ record/ xml. 2005 - 7
13 Termier A, Rousset M C, Sebag M. TreeFinder: A First Step Towards XML Data Mining. In: Proc. of the 2002 IEEE Int Conf on Data Mining, Maebashi TERRSA, Maebashi City, Japan, December 2002. 450～257
14 潘有能. XML 文档自动聚类研究. 情报学报, 2006, 25(2): 215～220

(上接第 141 页)

(2)系统根据课程名找到该课程概念，然后根据部分关系 P(概念 1，概念 2)和适应层次属性 ALevel(概念)在语义网中扩展，直至与知识点连接的概念，结束扩展；

(3)根据导出结构生成课件，在课件中可以同时列出每个知识点的相关概念链接；

(4)教师对生成的课件进行增、删、改操作，得出最终稿。

结论 通过应用案例，按照知识点进行扩展的语义网提供了一整套描述知识资源、知识资源查找和定位以及自动生成知识资源应用案例的方法，增强了知识资源整体关系的描述能力，查找和定位知识资源也更加方便，知识资源应用案例的生成也更加简易、灵活。若将其应用于各类组织(企业、院校及科研结构等)的知识管理系统，必将提升人们知识获取、分享、分配和存取的能力，为知识的理解和利用提供了一种有效的途径。

参 考 文 献

1 王晓蓉. 知识管理中的语义网方法[J]. 情报技术, 2004, 5(24): 65～66
2 沈军, 顾冠群. 面向网络教学的互动式体系模型[J]. 东南大学学报, 2002, 32: 6～10
3 Berners-Lee T. Weaving the Web [M]. San Francisco, CA: Harper, 1999
4 Berners-Lee T, Hendler J, Lassila O. The Semantic Web [J]. Scientific American, 2001, 284(5): 34～43
5 Berners-Lee T, Hendler J. Publishing on the semantic Web [J]. Nature 410, 2001, 26(6): 1023～1024
6 刘柏嵩. 基于知识的语义网：概念、技术及挑战[J]. 中国图书馆学报(双月刊), 2003, 2(3): 18～21
7 Staab S. Ontologies' KISSES in standardization [J]. IEEE Intelligent Systems, 2002, 6(24): 70～79
8 曲敏, 冯志勇. 基于 OWL ontology 的制造业知识管理[J]. 制造业自动化, 2006, 28(1): 8～12
9 张屹, 祝智庭. 知识管理在现代远程教育中的应用研究[J]. 中国远程教育, 2003, 3(182): 17