基于用户浏览行为聚类 Web 用户*)

陈 敏 苗夺谦 段其国

(同济大学电子与信息工程学院 上海 201804) (教育部嵌入式系统和服务计算重点实验室)

摘 要 本文结合 Web 用户浏览行为的特点,提出了一种新的路径相似度的计算方法,在计算相似度时不仅把用户的浏览模式仅作为一种序列模式来考虑,还充分考虑了用户在网上浏览的时间因素。然后,把粗糙度的概念引入 Leader 聚类算法中,提出粗糙 Leader 聚类算法。最后,使用标准数据集进行了试验,证明基于此种相似度计算方法,应用粗糙 Leader 算法聚类 Web 用户的有效性。

关键词 Web 日志挖掘,聚类,相似度,粗糙度

Clustering Web Users Based on Users' Browsing Action

CHEN Min MIAO Duo-Qian DUAN Qi-Quo

(Department of Computer Science and Engineering, Tongji University, Shanghai 201804)

(The Key Laboratory of Embedded System and Service Computing, Ministry of Education)

Abstract A novel method to get similitude actions of Web users is proposed in this paper after taking into account the characteristics of users' browsing actions. The new similarity is defined according to not only the browsing pages but also the time when users browse Web pages. Then, the concept of rough approximations is introduced in Leader cluster algorithm and rough Leader cluster algorithm is suggested. Finally, the performance of the rough Leader cluster algorithm is tested and analyzed by benchmark based on the novel method to computing the similarities of the web users' access patterns.

Keywords Web usage mining, Clustering, Similarity, Rough approximations

1 引言

作为 Web 智能(Web Intelligence)[1]的一个子研究课题,Web 日志挖掘是一个颇具前景的研究领域。通过挖掘 Web 访问日志可以获得 Web 访问用户的信息需求,这对于更加合理的规划网站结构,对用户提供个性化服务,为电子商务网站经营者发现潜在的客户等,都提供了非常有价值的信息。目前针对 Web 日志的分析方法很多,聚类作为一种重要的数据挖掘技术,已经在 Web 日志挖掘中得到了广泛的应用。

然而,相比于在传统数据挖掘中,聚类技术在 Web 日志挖掘中的应用仍有不少值得探讨的问题。在 Web 日志挖掘中,计算用户访问路径的相似度是聚类过程中十分重要的步骤之一。到目前为止,用户访问路径间相似度的计算大部分都是基于集合之间的交集运算,如夹角余弦方法[2]或 Jaccard 相关系数计算法[3],基于非欧式距离的序列排列方法(SAM),多维序列排列方法[4]等。正如我们所知,用户访问路径是指用户在一段时间内依次访问的页面的集合,然而这些相似度计算方法或者没有把访问路径作为一种序列来考虑,或者没有考虑用户访问时的时间因素。

本文结合 Web 用户浏览行为的特点,提出了一种新的路径相似度的计算方法,在计算相似度时不是把用户的浏览模式仅作为一个序列模式来考虑,还充分考虑了用户在网上浏览的时间因素。然后,把粗糙度的概念引入 Leader 聚类算法中,提出粗糙 Leader 聚类算法。最后,使用标准数据集进行了试验,证明基于此种相似度计算方法,应用粗糙 Leader 算法聚类 Web 用户的有效性。

2 Web 访问路径间相似度的计算

2.1 Web 访问路径

原始的 Web 日志数据中都不同程度地存在着缺失、谬误等噪音数据,所以必须进行预处理。经过过滤、用户识别、会话识别等步骤,就得到了包含用户访问路径的日志文件,这里我们使用 DePaul 大学提供的标准数据集[6]。数据主要来自 DePaul CTI Web 服务器(http://www.cs.depaul.edu),数据的采集是随机抽取在 2002 年 4 月的两个星期中访问这个网站的用户。每个会话(访问路径)以如下所示的一行作为会话的开始(见表 1):SESSION #n (USER_ID = k),其中 n 表示会话序号,k 表示用户的 ID。在一个给定的会话中,每一行对应用户发出的一个页面请求,包含三个域:时间戳,请求页面和引用页面。时间戳域表示用户访问页面的时间相距 2002 年 1 月 1 日的秒数。请求页面域的地址是以相对于 DePaul CTI Web 服务器的相对地址形式表示的。引用页面域表示所请求的页面是由哪个页面点击进入的,不难看出,这个域是为表明这条访问路径是用户连续访问的结果。因此,前两个域相对于最后一个域而言,对于分析用户的访问路径更有价值。

*)本文得到国家自然科学基金(60475019)和教育部博士点基金(20060247039)项目资助。**陈 敏** 博士,主要研究方向为 Web 智能,粗糙集;**苗夺谦** 教授,博导,主要研究方向之一为人工智能、模式识别;**段其国** 博士,主要研究方向为 Web 智能,粗糙集。

表 1

SESSION #20949 (USER_ID = 5446)		
10178867	/calendar/calendar.asp	-
10178870	/authenticate/login.asp?section=mycti&title=mycti&urlahead=studentprofile/studentprofile	/calendar/calendar.asp
10178880	/cti/studentprofile/studentprofile.asp?section=mycti	/authenticate/login.asp?section=mycti&title=mycti&urlahead=studentprofile/studentprofile

2.2 访问路径相似度的计算

在聚类 Web 访问路径中计算相似度时，一般的相似度计算方法只是把访问路径作为一种序列来考虑，却没有充分考虑用户访问时的时间因素。本文在计算用户访问路径间的相似度时，提出一种新的计算方法，它不仅考虑到访问路径间是否存在相同的访问路径，而且充分考虑到用户访问路径的时间因素。新的相似度计算方法主要是基于以上的分析所得的结论：两条访问路径，若它们拥有的共同访问路径越多，并且停留在共同访问路径上的时间越接近，则这两条访问路径就越相似。

设 m 条用户访问路径集合 $U=\{S_1,S_2,\cdots,S_m\}$，其中每条路径 $S_i=\{\langle p_1,\cdots,p_n\rangle,\langle t_1,\cdots,t_n\rangle\}$，$\langle p_1,\cdots,p_n\rangle$ 表示 S_i 中用户依次访问的各页面，$\langle t_1,\cdots,t_n\rangle$ 表示用户访问相应页面的时间。假设两个访问路径 S_i 和 S_j 之间共同的部分和不同的部分分别用 $f_{common}(S_i,S_j)$ 和 $f_{different}(S_i,S_j)$ 表示，那么计算 S_i 和 S_j 之间的相似度实际上是求 $f_{common}(S_i,S_j)$ 在 $f_{common}(S_i,S_j)$ 与 $f_{different}(S_i,S_j)$ 之和中所占比率的大小(见式(1))。

再设 S_i 和 S_j 之间相同的访问路径为 $C^k(S_i,S_j)(k=1,2\cdots Num_{\max})$，$Num_{\max}$ 代表 S_i 和 S_j 之间相同的访问路径数目。t_i^{kb} 和 t_i^{ke} 分别表示 S_i 在第 k 个相同路径上的开始访问时间和结束访问时间；同理可知 t_j^{kb} 和 t_j^{ke}。S_i 和 S_j 在 $C^k(S_i,S_j)$ 上的浏览时间比值如式(2)所示。显然，当 S_i 和 S_j 在 $C^k(S_i,S_j)$ 上停留的时间间隔越接近，则 S_i 和 S_j 越相似，即 $Rt_{i,j}^k$ 越接近 1 时 S_i 和 S_j 越相似。因此可得 $f_{common}(S_i,S_j)$ 的计算公式(见式(3))。其中 $|C^k(S_i,S_j)|$ 表示公共路径 $C^k(S_i,S_j)$ 的长度。式(3)表示 S_i 和 S_j 之间共同的部分与它们间共同的页面数以及在共同浏览路径上停留的时间的比值成正比。

其实也可计算出 S_i 和 S_j 之间平均浏览时间的比值。设 t_i^b 和 t_i^e 分别表示 S_i 的开始访问时间和结束访问时间，同理可知 t_j^b 和 t_j^e。所以 S_i 和 S_j 的平均浏览时间 $Avet_i$ 和 $Avet_j$ 如式(4)所示。其中 $|S_i|$ 和 $|S_j|$ 分别表示 S_i 和 S_j 的长度。进而可得 S_i 和 S_j 之间平均浏览时间的比值 $AveRt_{i,j}$，如式(5)所示。随之可得 $f_{different}(S_i,S_j)$ 的计算公式，如式(6)所示，其中 $\max(|S_i|,|S_j|)-\sum_{k=1}^{Num_{\max}}|C^k(S_i,S_j)|$ 表示 S_i 与 S_j 之间不同的页面数。最后把式(3)和式(6)代入式(1)中，就可得出完整的访问路径间相似度的计算公式。

$$Sim(S_i,S_j)=\frac{f_{common(S_i,S_j)}}{f_{common}(S_i,S_j)+f_{different}(S_i,S_j)} \tag{1}$$

$$Rt_{i,j}^k=\begin{cases}\min\left(\dfrac{t_i^{ke}-t_i^{kb}}{t_j^{ke}-t_j^{kb}},\dfrac{t_j^{ke}-t_j^{kb}}{t_i^{ke}-t_i^{kb}}\right) & \text{当 } t_i^{ke}\neq t_i^{kb} \text{ 且 } t_j^{ke}\neq t_j^{kb} \text{ 时}\\ 1 & \text{其他}\end{cases} \tag{2}$$

$$f_{common}(S_i,S_j)=\sum_{k=1}^{Num_{\max}}|C^k(S_i,S_j)|\cdot rt_{i,j}^k \tag{3}$$

$$Avet_i=\frac{t_i^e-t_i^b}{|S_i|}(t_i^e\neq t_i^b)\qquad Avet_j=\frac{t_j^e-t_j^b}{|S_j|}(t_j^e\neq t_j^b) \tag{4}$$

$$AveRt_{i,j}=\begin{cases}\min\left(\dfrac{Avet_i}{Avet_j},\dfrac{Avet_j}{Avet_i}\right) & \text{当 } t_i^e\neq t_i^b \text{ 且 } t_j^e\neq t_j^b \text{ 时}\\ 1 & \text{其他}\end{cases} \tag{5}$$

$$f_{different}(S_i,S_j)=(\max(|S_i|,|S_j|)-\sum_{k=1}^{Num_{\max}}|C^k(S_i,S_j)|)\cdot AveRt_{i,j} \tag{6}$$

3 聚类算法

Leader 聚类算法是一种只需扫描一遍数据库就能得到聚类结果的快速聚类算法，它寻找出一系列被称为 leader 的数据对象作为簇的代表。为了使 Leader 算法更适用于 Web 日志挖掘，符合 Web 日志挖掘的特点——用户访问目标的不确定性，引入粗糙聚类中粗糙度[5]的概念，对传统的 Leader 算法进行改进。下面给出改进的 Leader 算法步骤。

输入：Web 日志文件，全局阈值 τ，粗糙度阈值 ζ

输出：Web 访问路径聚类

步骤 1：以任意一个数据对象为初始的 Leader 对象；

步骤 2：对于数据集中每个数据对象 CP_i，执行以下操作：

2.1：求出目前可得的 Leader 对象与当前 CP_i 的相似度，并按相似度由大到小的顺序对 Leader 对象进行排序，得到排序后的 Leader 对象集 $\bigcup_L^i=\{L_1^i,L_2^i,\cdots,L_m^i\}$（$m$ 为目前可得的 Leader 对象数目）。

2.2：若 $Sim(CP_i,L_1^i)<\tau$，CP_i 作为一个新的 Leader 对象。否则，执行以下操作：

a) 分配 CP_i 对象到以 L_1^i 为代表的簇中；

b) 对于 $k>1$，若 $\dfrac{Sim(CP_i,L_k^i)}{Sim(CP_i,L_1^i)}>\zeta$，分配 CP_i 对象到以 L_k^i 为代表的簇中；否则，结束。

4 试验结果

试验中我们仍然使用 DePaul 大学提供的标准数据集[6]，并使用改进后的 Leader 算法聚类 Web 日志，在进行聚类时，采用的是从访问页面和访问时间两个角度考虑的访问路径相似度计算方法(设为 NewSim)。在算法实施时，我们发现 NewSim 与基于集合之间的交集运算计算所得的相似度(设为 OldSim $(S_i,S_j)=\dfrac{|S_i\cap S_j|}{|S_i\cup S_j|}$)有所不同。有些用 Oldsim 计算出的相似度比较高的访问路径在用 NewSim 计算时，相似度变低了；而有些用 Oldsim 计算出的相似度比较低的访问路径会用 Newsim 计算出较高的相似度。应该指出的是，Newsim 在计算相似度时，多从访问时间的角度来考虑，计算相似度更合理。

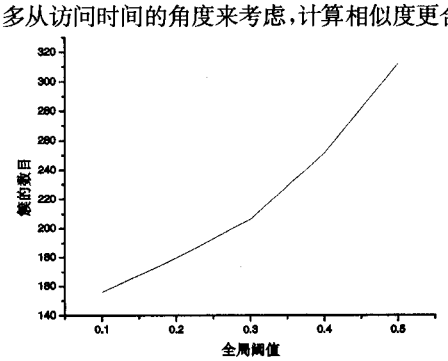

图 1 全局阈值 τ 与簇的数目的关系

(下转第 255 页)

我们选择三个大小为 512×512 的标准图像(包括 Baboon,Lena 和 Pepper)作为测试图像。实验中,SIFT 算法中用到的高斯函数的尺度范围是[2,32];相邻尺度之间的比率为$\sqrt{2}$;设置 $\lambda=15$。另外,实验中发现,特征尺度过小或过大时,相应的特征点不太稳定,因此在进行特征点的密度控制前,我们依据经验,从侯选特征点中剔除了特征半径小于 2、大于 10 的点。

原始 Baboon,Lena 和 Pepper 图像上生成的 Delaunay 三角网的三角形数量分别是 101、70 和 59。原始 Baboon,Lena 和 Pepper 图像上生成的 Delaunay 三角网分别如图 3 所示。信号处理和几何攻击后的图像上生成的 Delaunay 三角网的三角形的重复率 R 分别如表格 1 和 2 所示。

表 1 信号处理攻击下的重复率 R

攻击	Baboon	Lena	Pepper
Median filter 2×2	13.9	15.7	20.3
Median filter 3×3	10.9	11.7	10.2
FMLR	16.8	5.7	3.4
Gaussian filter 3×3	20.8	11.4	30.5
JPEG 90	49.5	54.3	49.2
JPEG 70	32.7	35.7	44.1
JPEG 50	27.7	24.3	37.3

表 2 几何攻击下的重复率 R

攻击	Baboon	Lena	Pepper
Cropping 20% off	23.8	8.5	15.3
Rotation 10	9.9	7.1	6.8
Scaling 75%	5.0	12.9	1.7
Scaling 150%	5.9	11.4	11.9
Aspect ratio change (1.10,1.00)	8.9	8.6	11.9
Aspect ratio change (1.00,1.10)	6.9	17.1	15.3

从表中数据可以看出,总地来说,图像 Baboon 的区域的重复率 R 不及图像 Lena 和 Pepper,这主要是由于图像 Baboon 的纹理性比较强。而且,当遭受轻微的纵横比改变时,仍有少量的局部区域被正确检测到。总地来说,用于同步水印的大多数不变区域对于一般的信号处理攻击和几何攻击是鲁棒的。而且,只要攻击后的图像上至少一个用于嵌有水印的局部区域被正确检测到,则可判断被检测的图像带有版权信息。因此,总地说来,SIFT 特征点用于同步水印是可行的。

结论和未来展望 目前的大多数鲁棒的数字水印算法不能抵抗几何攻击,根本原因是图像遭受几何攻击后,水印检测位置和水印嵌入位置的同步性会被破坏。局部图像水印算法,将水印以图像特征为参照点嵌入到各个局部区域中,从而水印的同步问题迎刃而解。

本文利用 SIFT 特征点来同步水印。实验数据表明,SIFT 特征点用于同步水印是可行的,由 SIFT 特征点生成的局部区域不仅对信号处理鲁棒,对于一般的几何攻击,包括裁剪、旋转、缩放、平移和轻微的纵横比改变等都具有较好的鲁棒性。下一步的工作是寻找鲁棒的大容量的水印算法,并将该算法用于各个局部区域上,实现一套完整的局部水印方案。

参 考 文 献

1 Sharma G, Coumou D J. Watermark Synchronization: Perspective and a New Paradigm. In: Proc. 40th Annual Conference on Information Sciences and Systems, NJ, March 2006. 1182～1187

2 Bas P, Chassery J-M, Macq B. Geometrically Invariant Watermarking Using Feature Points. IEEE Trans. Image Process, 2002,11(2):1014～1028

3 Mikolajczyk K, Schmid C. An affine invariant interest point detector. In: Proc. ECCV, vol. 2350, LNCS,2002. 128～142

4 Zhen J, Lai J, Jing J, et al. An Improved Second Generation Digital Image Watermarking Scheme. Mathematics of Data/Image Coding, Compression, and Encryption VI, with Applications, 2004, 5208:218～223

5 Lowe D G. Dinstinctive image features from scale-invariant keypoints. International Journal of Computer Vision,2004, 60: 91～110

6 Bertin E, Marchand-Maillet S, Chassery J -M. Optimization in voronoi diagrams. Norwell, MA: Kluwer, 1994. 209～216

(上接第 187 页)

为了表明改进后的 Leader 算法的性能,图 1 给出了全局阈值 τ 与簇的数目的关系,图 2 给出了粗糙阈值 ζ 与边界对象数目的关系。试验结果表明,改进后的 Leader 算法通过设定粗糙阈值 ζ,可以控制边界对象的数目,结合全局阈值 τ 的变化,可以实现对 Web 访问路径的聚类,使得每个簇有确定的表示,并且访问目的不确定的用户可以同时属于多个簇,实现粗糙分类。

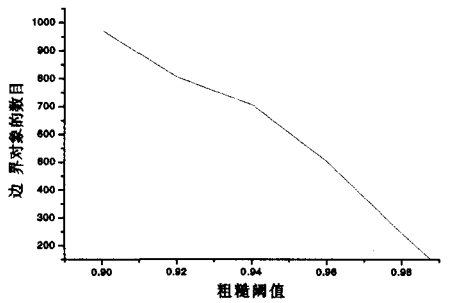

图 2 粗糙阈值 ζ 与边界对象数目的关系

参 考 文 献

1 Zhong N, Liu J, Yao Y Y. Special issue on Web Intelligence (WI). Hingham,MA,USA: Kluwer Academic Publishers,2003

2 Fayyad M, Piatetsky-Shapiro G, Smyth P. From data mining to Knowledge discovery: an overview. In: Fayyad M, Piatetsky-Shapiro G, Smyth P, eds. Advances in knowledge Discovery and Data Mining, Menlo Park, CA: AAI. Press, 1996. 1～36

3 Foss A, Wang W, Zaïane O R. A Non-Parametric Approach to Web Log Analysis. In : Proceedings of Workshop on Web Mining in First International SIAM Conference on Data Mining , 2001

4 Hay B, Wets G, Vanhoof K. Web Usage Mining by Means of Multidimensional Sequence Alignment Methods. Lecture Notes in Computer Science, 2003, 2703: 50～65

5 Peters G. Some refinement of K-means clustering. Pattern Recognition, 2006, 39: 1481～1491

6 http://maya.cs.depaul.edu/～classes/ect584/resource.html