

文章编号:1001-9081(2009)10-2774-04

基于 Web 日志的典型匿名用户路径挖掘研究

缪 勇¹, 宋 斌²

(1. 扬州环境资源职业技术学院 计算机科学技术系, 江苏 扬州 225127; 2. 南京理工大学 计算机科学与技术系, 南京 210094)
(mmm_01@163.com)

摘 要:通过获取的匿名用户浏览路径集,依据新的路径相似度定义,建立用户浏览路径相似度矩阵,并在此基础上设计实现了匿名用户浏览路径聚类算法,获得聚类结果集,并计算各类的中心,得到典型匿名用户路径。挖掘结果显示典型匿名用户路径代表了不同类用户网络浏览路径,可有效地作为网站信息推荐的依据。

关键词:Web 日志; 匿名用户; 相似度; 路径聚类; 典型匿名用户浏览路径

中图分类号: TP311 **文献标志码:** A

Research on mining typical anonymous users' browsing paths based on Web logs

MIAO Yong¹, SONG Bin²

(1. Department of Science and Technology, Yangzhou Vocational College of Environment And Resources, Yangzhou Jiangsu 225127, China;
2. Department of Computer, Nanjing University of Science and Technology, Nanjing Jiangsu 210094, China)

Abstract: Through obtaining anonymous users' browsing path set, on the basis of the new definition of path similarity, building the users' browsing path similarity matrix, and based on this, designing anonymous users' browsing path clustering algorithm to obtain the clustering result set and calculate various cluster centers, the typical anonymous users' browsing path was got. Experimental results show that a typical anonymous users' browsing path represents different types of users' Web browsing path, which can be effective reference for Web site information recommendation.

Key words: Web log; anonymous user; similarity; path clustering; typical anonymous users' browsing path

用户的浏览信息通常由 Web 服务器自动收集,以 Web 日志的形式存储。利用 Web 日志进行用户浏览路径挖掘是 Web 挖掘的一个研究方向,其挖掘结果可以分析改进网络性能、优化 Web 站点的拓扑结构以及进行个性化服务等^[1-2]。实际应用时最终的用户模式需要依赖 Cookies 信息等技术完成,通常情况下很难获得准确的 Cookies 信息从而无法对用户进行准确的描述,这种在无明确的用户信息情况下的用户模式挖掘称为匿名用户数据挖掘。由于匿名用户的数量巨大,因此基于 Web 日志的匿名用户路径挖掘具有重要意义。

1 Web 日志数据预处理

对 Web 日志数据预处理是在将日志文件导入数据库以后进行的,其目的是把 Web 日志转化为适合进行典型匿名用户路径挖掘的可靠和精确的数据。包括几个阶段:数据净化、格式转换、用户识别和会话识别。

1.1 数据净化

数据净化指从 Web 日志中过滤出研究所关心的数据,由于以获得用户的行为模式为目标,因此不必考虑用户无显式请求的有关文件,可采用检查 URL 后缀、服务器返回代码等方法,剔除掉访问图像文件条目、访问失败条目以及 CGI 为后缀的脚本文件和张贴方法所产生的条目。经过数据净化,Web 日志文件在数据量上通常只有原始 Web 日志文件的 1/10~1/4。

1.2 格式转换

为便于以后的进一步处理,需确定适当的数据表示形式,为此本文对日志记录中的访问页 URL 进行了编号。具体处

理时,可以根据站点拓扑结构进行编号,但存在每当站点结构发生变化时,都要重新获得站点结构的缺点,而站点结构是经常变化的,所以这种编号方法很不方便。因此本文采用一种较简单的方法,利用 SQL 语句,在经过数据清洗的日志数据表 logdata 中选出用户访问过的不同的 URL,然后按顺序编号即可。

1.3 用户识别

一般采用以下启发式规则来识别匿名用户^[3]:

- 1) 不同的 IP 地址代表着不同的用户;
- 2) 当 IP 地址相同时,默认不同的操作系统或浏览器代表不同的用户;
- 3) 在 IP 地址相同,用户使用的操作系统和浏览器也相同的情况下,则需根据网站的拓扑结构图对用户进行识别。

1.4 会话识别

用户会话是指用户对服务器的一次有效访问,通过其连续请求的页面,可获得其在网站中访问行为和浏览兴趣。Web 日志中不同用户访问的页面属于不同的会话。某个用户的页面请求在时间上跨度比较大时,可能是该用户多次访问同一个网站,因此可将用户的访问记录分成多个会话,通常使用时间阈值法来处理,同一个用户访问的页面序列中,若相邻的 2 个页面的时间差超过给定的阈值,就将其分到不同的会话序列中。

2 基于 Web 日志的典型匿名用户路径挖掘

2.1 典型匿名用户路径挖掘流程

典型匿名用户路径的挖掘包括获取用户浏览路径集、计

收稿日期:2009-04-16;修回日期:2009-06-08。

基金项目:国家自然科学基金资助项目(60850002);南京理工大学科研发展基金(A96134)。

作者简介:缪勇(1975-),男,江苏句容人,讲师,硕士,主要研究方向:数据挖掘、Web 信息处理; 宋斌(1968-),男,江苏南京人,副教授,主要研究方向:数据挖掘、Web 信息处理、网络技术与应用、网络信息处理。

算浏览路径间相似度、对用户浏览路径进行聚类 and 计算聚类结果集中各类的中心四个步骤,具体流程如图 1 所示。

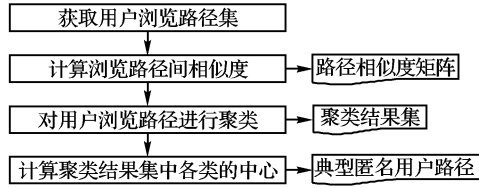


图 1 典型匿名用户路径挖掘流程

2.2 用户浏览路径集的获取

数据预处理阶段完成用户会话识别后,对每个会话,只保留第一次出现的 URL 地址,如果出现重复,则删除,经过处理后,便可获得全体用户在一个时间段内对站点的浏览路径集,包含 n 条用户浏览路径的集合可表示为 $S = \{s_1, s_2, \dots, s_n\}$ 。得到了用户浏览路径集,就可以计算路径之间的相似度。

2.3 建立用户浏览路径相似度矩阵 M_{sim}

定义 1 $|s|$ 是用户浏览路径 s 的长度,即路径 s 中用户浏览的页面数。

计算两个用户浏览路径 s_i, s_j 的相似度要考虑以下三种情况。

1) 不考虑用户浏览路径中页面的顺序时,如式(1):

$$sim_1(s_i, s_j) = |s_i \cap s_j| / |s_i \cup s_j| \quad (1)$$

其中: $|s_i \cap s_j|$ 为两条浏览路径中相同页面数, $|s_i \cup s_j|$ 为两条浏览路径总页面数。如有 2 条用户浏览路径 $s_1 = \{P1, P2, P3\}, s_2 = \{P1, P3, P2\}$, 显然这两条路径不是同一条路径,但根据式(1)计算, s_i 和 s_j 的相似度为 1。因此,只从这一种情况考虑路径相似度并不准确。

在一定程度上考虑用户浏览路径中页面的顺序,如式(2):

$$sim_2(s_i, s_j) = \sum_{k=1}^m t_k(s_i, s_j) / \max(|s_i|, |s_j|) \quad (2)$$

其中:

$$t_k(s_i, s_j) = \begin{cases} 1, & \text{浏览路径 } s_i \text{ 与 } s_j \text{ 中第 } k \text{ 个浏览页面相同} \\ 0, & \text{浏览路径 } s_i \text{ 与 } s_j \text{ 中第 } k \text{ 个浏览页面不同} \end{cases}, k = 1, 2, \dots, m$$

其中: $\max(|s_i|, |s_j|)$ 表示路径 s_i 和 s_j 中最长路径的长度, m 为 s_i 和 s_j 中最小路径的长度。

若有 3 条用户浏览路径 $s_1 = \{P1, P2, P5\}, s_2 = \{P1, P3, P5\}, s_3 = \{P1, P2, P6\}$, 则根据式(2)计算, 路径 s_1 和 s_2 的相似度为 2/3, 路径 s_1 和 s_3 的相似度也是 2/3。事实上, 路径 s_1 和 s_3 应该比 s_1 和 s_2 更相似。所以, 只从这种情况考虑路径相似度也并不准确。

2) 完全考虑用户浏览路径中页面的顺序, 如式(3):

$$sim_3(s_i, s_j) = |com(s_i, s_j)| / \max(|s_i|, |s_j|) \quad (3)$$

其中: $|com(s_i, s_j)|$ 表示路径 s_i 和 s_j 中最大相同路径的长度, $\max(|s_i|, |s_j|)$ 表示路径 s_i 和 s_j 中最长路径的长度。

如有 3 条用户浏览路径 $s_1 = \{P1, P2, P4, P5\}, s_2 = \{P1, P2, P3, P6\}, s_3 = \{P1, P2, P5, P4\}$, 则 s_1, s_2 的最大相同路径为 $P1 - P2$, 长度为 2, 根据式(3)计算, 路径 s_1, s_2 的相似度为 0.5, s_1, s_3 的最大相同路径为 $P1 - P2$, 长度为 2, s_1, s_3 的相似度计算结果也为 0.5。实际上, 路径 s_1, s_3 中的页面完全相同, 大致次序也相同, 其实际的相似度应该大于 s_1, s_2 的相似度。

为此, 本文将以上三种情况综合考虑, 给出计算路径相似度的定义。

定义 2 两个用户浏览路径 s_i, s_j 的相似度 $= \alpha \times sim_1(s_i, s_j) + \beta \times sim_2(s_i, s_j) + \gamma \times sim_3(s_i, s_j), \alpha + \beta + \gamma = 1$ 。

α, β, γ 为调节系数, 通过设置 α, β 和 γ 的大小, 可以改变 $sim_1(s_i, s_j), sim_2(s_i, s_j)$ 和 $sim_3(s_i, s_j)$ 在计算路径相似度时所起的作用。

完成用户浏览路径集中所有路径间的相似度计算后, 便可得到如下所示的用户浏览路径相似度矩阵 M_{sim} 。该矩阵是一个上三角矩阵。

$$M_{sim} = \begin{matrix} & s_1 & s_2 & s_3 & s_4 & \cdots & s_n \\ \begin{matrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ \vdots \\ s_n \end{matrix} & \begin{bmatrix} 1 & s_{12} & s_{13} & s_{14} & \cdots & s_{1n} \\ & 1 & s_{23} & s_{24} & \cdots & s_{2n} \\ & & 1 & s_{34} & \cdots & s_{3n} \\ & & & 1 & \cdots & s_{4n} \\ & & & & \ddots & \\ & & & & & 1 \end{bmatrix} \end{matrix}$$

2.4 基于用户浏览相似度矩阵的聚类算法

由于用户浏览相似度矩阵维数很高, 占用内存空间很大的, 一般算法很难对其聚类, 本文采用了一种聚类方法^[6] (User Browsing Path Clustering, UBPC), 该算法适于处理高维数据, 具有较好的实用性, 可以解决大数据量的 Web 用户浏览路径聚类。

1) UBPC 算法思想描述

对用户浏览路径集 $S = \{s_1, s_2, \dots, s_n\}$ 中的每一个 s_i 根据给定阈值 θ 构造 s_i 的相似类 $[s_i]_R, [s_i]_R = \{s_j | s_j \in S, s_{ij} \geq \theta\}^{[4-5]}$ 。

由于用户浏览路径相似度矩阵不满足传递性, 因此, 最初得到的只是相似类而不是等价类, 即对每一个 $s_i \in S$ 来说, 在得到的若干个相似类中可能有重复类或各类之间存在相交项。因此, 需要删除重复的相似类和各个相似类之间的相交项, 最终得到的等价类即为聚类的结果。

2) UBPC 聚类算法

输入: 用户浏览路径相似度矩阵 M_{sim} , 路径相似度阈值 θ

输出: 聚类结果集 $PCS = \{PCS_k\}, k = 1, \dots, m, PCS_k$ 是 PCS 中的第 k 个类

```

m = 0 //m 为聚类结果集 PCS 中当前的聚类数目
for ( i = 0; i < n; i++) // n 为矩阵 M_sim 的行数
{
    CS = {}; //CS 存放临时类
    for ( j = i; j < n; j++)
    {
        if ( M_sim[i][j] >= theta )
            将路径编号 s_{j+1} 加入临时类 CS 中;
    }
    // 判断 CS 是否是 PCS 中某个类的子集
    IsIncluded = false;
    for ( k = 1; k <= m; k++)
        //m 为当前 PCS 中聚类的数目, PCS_k 为 PCS 中第 k 个类
    {
        if ( CS 是 PCS_k 的子集 )
        {
            IsIncluded = true;
            break;
        }
    }
    //如果 CS 不是 PCS 中任何类的子集, 将 CS 加入到 PCS 中
    if ( IsIncluded == false )
    {
        m++;
        PCS_m = CS;
    }
}
//消除聚类结果集 PCS 中各个类之间的相交项
for ( i = 1; i < m; i++)

```

```

{
for(  $j = i + 1; j \leq m; j++$  )
{
     $IntersectItemSet = PCS_i \cap PCS_j$ ;
    //IntersectItemSet 为两个类的相交项
    for each  $x \in IntersectItemSet$ 
    {
        //计算相交项  $x$  分别隶属于类  $PCS_i$  和  $PCS_j$  的程度,从隶属度小的类中删除  $x$ 
        if (  $Cal(x, PCS_i) \geq Cal(x, PCS_j)$  )
            从类  $PCS_j$  中将  $x$  项删除;
        else
            从类  $PCS_i$  中将  $x$  项删除;
    }
}
}
//函数  $Cal(x, PCS_i)$  计算  $x$  隶属于类  $PCS_i$  的程度
 $p$  = 类  $PCS_i$  中用户浏览路径数;
 $sum = 0$ ;
for(  $k = 1; k \leq p; k++$  )
{
    if(  $PCS_i[k] \neq x$  )
        //  $PCS_i[k]$  为类  $PCS_i$  中第  $k$  个用户浏览路径
         $sum = sum +$  路径  $x$  与路径  $PCS_i[k]$  的相似度;
}
return  $sum / (p - 1)$ ;

```

2.5 典型匿名用户路径的获取

用户浏览路径通过聚类,得到的每个类中通常包含多条用户浏览路径,可以在各个类中寻找一条具有代表性的路径,即各个类的中心。

定义 3 聚类的中心为该聚类中所有用户浏览路径的最大相似路径。假设 PCS_i 为聚类结果集 PCS 中的一个类, PCS_i 的中心 = $\{url_1, \dots, url_k, \dots, url_m\}$ 。其中 url_k 为类 PCS_i 中各条用户浏览路径第 k 次所浏览的页面中浏览次数最大的页面,且该页面不属于 $\{url_1, url_2, \dots, url_{k-1}\}$, 如果该页面属于 $\{url_1, url_2, \dots, url_{k-1}\}$, 则考虑浏览次数次大的页面。

计算出各个类的中心,便得到了典型匿名用户路径。

2.6 典型匿名用户路径挖掘实例

假设有如下 6 条用户浏览路径: $s_1: p1-p2-p7, s_2: p1-p3-p7, s_3: p1-p2-p8, s_4: p1-p3, s_5: p1-p5-p8, s_6: p1-p4-p7$

则 $sim_1(s_1, s_2) = |\{p1, p7\}| / |\{p1, p2, p3, p7\}| = 1/2$, $sim_2(s_1, s_2) = 2/3, sim_3(s_1, s_2) = |com(s_1, s_2)| / \max(|s_1|, |s_2|) = |\{p1\}| / 3 = 1/3$, 取 $\alpha = 0.2, \beta = 0.3, \gamma = 0.5$, 则路径 s_1 和 s_2 间的相似度 $sim(s_1, s_2) = \alpha \times sim_1(s_1, s_2) + \beta \times sim_2(s_1, s_2) + \gamma \times sim_3(s_1, s_2) = 0.2 \times 1/2 + 0.3 \times 2/3 + 0.5 \times 1/3 = 0.46$ 。

其他路径间的相似度计算类似,则得到如下的用户浏览路径相似度矩阵 M_{sim} 。

$$M_{sim} = \begin{matrix} & \begin{matrix} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \end{matrix} \\ \begin{matrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{matrix} & \begin{bmatrix} 1 & 0.46 & 0.63 & 0.32 & 0.31 & 0.46 \\ & 1 & 0.31 & 0.66 & 0.31 & 0.46 \\ & & 1 & 0.32 & 0.46 & 0.31 \\ & & & 1 & 0.32 & 0.32 \\ & & & & 1 & 0.31 \\ & & & & & 1 \end{bmatrix} \end{matrix}$$

用 UBPC 聚类算法进行聚类,取阈值为 0.4,在执行消除聚类结果集中各类的相交项之前,得到的聚类结果集 PCS 中 3 个类为: $PCS_1 = \{s_1, s_2, s_3, s_6\}, PCS_2 = \{s_2, s_4, s_6\}, PCS_3 = \{s_3, s_5\}$ 。

类 PCS_1 和 PCS_2 之间存在相交项 s_2 和 s_6 ,算法在执行删除相交项时计算出 s_2 隶属于类 PCS_1 程度 = $(M_{sim}[s_1][s_2] + M_{sim}[s_2][s_3] + M_{sim}[s_2][s_6])/3 = (0.46 + 0.31 + 0.46)/3 = 0.41$, s_2 隶属于类 PCS_2 程度 = $(M_{sim}[s_2][s_4] + M_{sim}[s_2][s_6])/2 = 0.56$ 。因为 s_2 隶属于类 PCS_2 程度要大于 s_2 隶属于类 PCS_1 程度,所以类 PCS_1 中的项 s_2 被删除,得到的中间结果为: $PCS_1 = \{s_1, s_3, s_6\}, PCS_2 = \{s_2, s_4, s_6\}, PCS_3 = \{s_3, s_5\}$ 。

相交项 s_6 隶属于类 PCS_1 程度 = $(M_{sim}[s_1][s_6] + M_{sim}[s_3][s_6])/2 = 0.385$, 相交项 s_6 隶属于类 PCS_2 程度 = $(M_{sim}[s_2][s_6] + M_{sim}[s_4][s_6])/2 = 0.39$, 因此将类 PCS_1 中的 s_6 删除,得到的中间结果为: $PCS_1 = \{s_1, s_3\}, PCS_2 = \{s_2, s_4, s_6\}, PCS_3 = \{s_3, s_5\}$ 。

类 PCS_1 和 PCS_3 之间存在相交项 s_3 , 计算 s_3 隶属于类 PCS_1 程度 = $M_{sim}[s_1][s_3] = 0.63$, s_3 隶属于类 PCS_3 程度 = $M_{sim}[s_3][s_5] = 0.46$, 所以将类 PCS_3 中的 s_3 删除,得到的结果为:

$$PCS_1 = \{s_1, s_3\}, PCS_2 = \{s_2, s_4, s_6\}, PCS_3 = \{s_5\}。$$

此时,各类之间已无相交项,聚类算法结束。计算各类的中心获得典型匿名用户路径为: $p1-p2-p7$ (或 $p1-p2-p8$), $p1-p3-p7, p1-p5-p8$ 。

取阈值为 0.32 时,删除相交项前结果为: $PCS_1 = \{s_1, s_2, s_3, s_4, s_6\}, PCS_2 = \{s_3, s_4, s_5\}, PCS_3 = \{s_4, s_5, s_6\}$ 。删除相交项后,得到的结果为: $PCS_1 = \{s_1, s_2, s_3, s_4, s_6\}, PCS_2 = \{s_5\}$ 。

计算各类的中心获得的典型匿名用户路径为: $p1-p2-p7$ (或 $p1-p3-p7$), $p1-p5-p8$ 。

2.7 实验结果

实验环境使用的机器为: Intel (R) Celeron (R) M processor 1.40 GHz, 512 MB 内存。开发平台中操作系统为 Windows XP, 利用 C# 进行开发, 后台使用的数据库为 SQL Server 2000。实验数据来源于南京理工大学 (<http://www.njust.edu.cn>) Web 服务器日志文件。选择一个大小为 30.1 M 的 Web 日志记录导入数据库中,共得到 166 186 条记录,经过数据清洗,删除后缀为 jpg, jpeg, gif, ico, js, css 等的记录,得到 15 016 条记录,数据记录缩小了 10 倍多。从清洗后日志记录表中识别出 934 个用户。以 30 min 作为 timeout,从上述 934 个用户中识别出 1 356 条会话。

通过对用户会话文件的处理获得用户浏览路径集,该集合中有 1 157 条用户浏览路径,删除相同的路径后得到 387 条不同的用户浏览路径。

对上述 387 条不同用户浏览路径进行聚类,实验结果如表 1 所示。

表 1 相似度阈值大小与得到的聚类数关系

相似度阈值	聚类数
0	1
0.2	67
0.4	235
0.5	264
0.6	297
0.7	342
0.8	372
1.0	387

相似度阈值大小与聚类数变化关系如图 2 所示。图 2 表

