基于 R 树的协同过滤推荐算法①

张龙昌, 张洪锐

(渤海大学 信息科学与技术学院, 锦州 121000)

摘 要: R 树是一个高度平衡树, 也是目前应用最为广泛的空间索引结构. 本文以用户行为的历史数据之间的相似度构造 R 树, 提出一种基于 R 树的协同过滤推荐算法(R_CF); 另外, 从用户的隐式反馈着手, 构建用户兴趣行为数据模型, 并进行数据标准化处理. 仿真实验表明: 较之传统的协同过滤推荐算法(CF), 本文提出的 R_CF 算法可以极大提升推荐 top-n 个相似度最高的用户时的查询速度.

关键词: R 树; 协同过滤推荐算法; 隐式反馈; 用户兴趣行为数据模型

Collaborative Filtering Recommendation Algorithm Based on R Tree

ZHANG Long-Chang, ZHANG Hong-Rui

(College of Information Science and Technology, Bohai University, Jinzhou 121000, China)

Abstract: The R-tree is a highly balanced tree, and is the most widely used spatial index structure. Based on the similarity of the historical data, this paper constructs R-tree, and proposes a collaborative filtering recommendation algorithm based on R-tree (R_CF). In addition, this paper sets about from the user's implicit feedback, builds the user interest behavior data model, and standardizes data. Simulation experiments show that compared with the traditional collaborative filtering recommendation algorithm (CF), the proposed R_CF algorithm can greatly improve recommended top-n query speed.

Key words: R-tree; collaborative filtering recommendation algorithm; implicit feedback; user interest behavior data model

随着电子商务网站的发展, 产品和用户数增长迅速, 相应的大数据处理技术也不断涌现, 其中, 如何帮助用户过滤、筛选出自己喜好的产品和服务, 是提高用户满意度和增加企业效益的关键. 所以, 个性化推荐算法也要适应大数据时代的这个形势, 不断提高推荐的命中率和效率.

本文针对传统的协同过滤算法, 在相似度的存储结构上做出一些改进, 提出一种基于 R 树的协同过滤推荐算法. 用户的历史行为数据是研究推荐算法的关键资源, 具体分为两类: 显式反馈和隐式反馈[1]. 显式反馈是用户对购买、使用的商品做出评分等主观感受反馈; 隐式反馈则是用户在浏览、购买商品时产生的一系列客观行为, 如收藏商品、点击连接等. 本文从用户的隐式反馈着手, 构建用户兴趣行为历史数据模型, 作为推荐算法的研究基础.

1 数据模型

定义 1(用户兴趣行为). 用户浏览网站过程中, 服务器会记录一系列的浏览行为, 如拉动滚动条, 点击链接等, 浏览器会记录页面保存于书签中等, 这些浏览行为都能体现用户对当前浏览内容的感兴趣, 这些浏览行为统称为用户兴趣行为[2]. 记为:

$$UIB=\{b_i,b_2,\cdots,b_n\}$$

相对而言, 不感兴趣的浏览行为包括由于浏览器崩溃导致关闭或者重启浏览器等行为, 这些行为就不会作为收集的感兴趣行为数据集中. 本文选取几个有

① 基金项目:辽宁省教育厅科学技术研究一般项目(L2014451)

收稿时间:2016-02-27;收到修改稿时间:2016-03-31 [doi:10.15888/j.cnki.csa.005430]

代表性的用户兴趣行为(1)作为本文算法的原始数据集. 一个用户一次访问过程中的浏览行为的表示形式如式(1).

$$UBL = < uip, wip, url, title, < time, behave >> \quad (1)$$

uip 是用户的 ip 地址, 一个用户的唯一标示, *wip* 是服务器 ip 地址, 唯一标识一台服务器, *url* 是当前用户浏览的链接地址(即用户访问的感兴趣的资源内容), *title* 是记录当前 *url* 的标题内容, 用于向访问行为相似的用户推送感兴趣内容的标题. $< time, behave >$ 是一个数组, 代表 *time* 时刻 *uip* 用户正在做 *behave* 动作, 其中 $behave \in UIB$.

表 1　用户兴趣行为规格化

用户兴趣行为	度量标准	描述
保存、删除标签 b_1	$\{0,5,10\}$	0-未保存标签, 5-删除标签, 10-保存到标签
打印页面 b_2	$\{0,10\}$	0 代表未打印, 10 代表打印
查询关键词 b_3	$[0,10]$	关键词个数, 按式(2)规格化为 0-10 区间
点击链接 b_4	$[0,10]$	点击链接数, 按式(2)规格化为 0-10 区间
平均浏览时间 b_5	$[0,10]$	按式(2)将时间规格化为 0-10 区间内
拉动滚动条 b_6	$[0,10]$	拉动滚动条的次数, 按式(2)规格化 0-10 区间

2　web日志数据处理

数据预处理是首要的并且是很重要的一步. Web 日志的数据来源主要是服务器端日志、代理服务器端日志和客户端日志[3,4]. 这些数据有噪声、冗余、不一致等一系列问题. 如果不针对这些问题对原始数据进行清洗、规格化等预处理操作, 就会导致数据挖掘出来的知识会很不准确. 所以, 本节首先简要介绍一下预处理需要做的数据清理、数据集成、数据变换等步骤, 并描述了记录用户行为的数据模型(第 1 节).

数据预处理一般分为数据清理、数据集成、数据变换和数据规约四步. 数据清理工作往往是针对 web 日志中产生的一些图片、音频等文件, 所以需要将后缀为 gif、jpg 等文件删除以达到格式标准化; 数据集成是将多个数据源中的数据集成到一起统一存储, 例如本文设计的推荐模型就是将来自服务器端日志、代理服务器端日志和客户端日志三个不同来源的日志经过预处理后的数据存储到数据仓库中; 数据变换是将属性数据按照比例缩放到一个统一区间内, 这部分工作对于基于距离的挖掘算法尤为重要, 所以在 2.1 节中着重介绍.

2.1 数据标准化

设 $Y=(y)_{m\times n}$ 为表 1 中用户兴趣行为的决策矩阵, y_{ij} 表示第 i 个用户的第 j 个行为的数据采集值(b_1 和 b_2 的行为的采集值分别是 $\{0,0.5,1\}$ 、 $\{0,1\}$). 令 $Z=(z)_{m\times n}$ 为进行数据标准化后的矩阵, 同理, z_{ij} 表示第 i 个用户的第 j 个行为的数据标准化处理后的值. $Y=(y)_{m\times n}$ 转化成 $Z=(z)_{m\times n}$ 的公式如式(2).

$$z_{ij}=k\left(\frac{y_{ij}-y_j^{\min}}{y_j^{\max}-y_j^{\min}}\right) \quad (2)$$

其中, $y_j^{\min}$ 是 $Y=(y)_{m\times n}$ 中 j 列的最小值, $y_j^{\max}$ 是 $Y=(y)_{m\times n}$ 矩阵中 j 列的最大值, 也就是这 m 个用户中第 n 种用户兴趣行为的最小值和最大值, k 是各个属性的权重比例, 本文中取值为 10.

3　基于R树的协同过滤推荐算法

R 树是一个高度平衡树, 也是目前应用最为广泛的空间索引结构, 所有的有效数据都存储于叶子结点, 非叶子结点只是作为一种索引, 从而可以优化检索效率. 传统的协同过滤算法是对目标用户与历史用户兴趣行为进行两两计算, 对与目标用户的相似度进行排序, 以此获取 top-n 个用户进行推荐. 本节提出的基于 R 树的协同过滤算法(R_CF)是通过用户兴趣行为之间的相似度来进行构建树结构, 对用户兴趣行为进行标准化处理后, 以用户兴趣行为的相似度作为索引, 能够快速获取与目标用户相似的 top-n 个用户, 从而提高数字图书馆云服务推荐速度. 本节 3.1 首先介绍了 R 树的定义及构造原理, 然后 3.2 节举例介绍了 R_CF 算法.

3.1 R 树构造

基于 *R* 树空间索引结构存储数据, 通过设置结点包含孩子结点的最大数 M 和最小数 $M/2$, 超过最大数 M 就自动分裂为两个孩子结点, 这样就使得分裂后的结点仍然保持孩子结点的最大数 M 和最小数 $M/2$ 之间. *R* 树空间索引结构的索引即为用户兴趣行为向量的相似度. 在构造 *R* 树的过程中, 判断新增的用户兴趣行为插入 *R* 树的哪一个叶子结点, 依据的是索引空间的扩张的最小, 也就是把最相似度最接近的用户兴趣行为组合, 放在同一个结点中[5].

另外, R 树的每一层都实现了对有效数据的聚类: 对于叶子结点层来说, 由于叶子结点之间相互不会重叠, 所以每一个有效数据都唯一属于某一类, 类似于聚类算法中的硬聚类; 每一个非叶子结点可能会与其他非叶子结点有重叠, 因此, 每一个非叶子结点(R 树同一层)包含的有效数据同样属于一类, 但一个有效数据可能属于多个同一层中非叶子结点, 这也类似于聚类中的软聚类(也就是模糊聚类). 下面定义 2 描述了构建用户兴趣行为 R 树的结构. 算法 1 是 R 树的构造算法[6,7].

定义 2(R 树). R 树结点 *entity* 所保存的数据形式为 $(R_{\min}, R_{dis}, parent, childList)$:

1) $R_{\min}$ 是 n 维空间的一个点, $R_{\min}=\left(R_1^{\min}, R_2^{\min}, \cdots, R_n^{\min}\right)$, 是此结点在任一维空间上的最小值组成的.

2) R_{dis} 对应的是一个 n 维空间向量, $R_{dis}=\left(R_1^{\max}-R_1^{\min}, R_2^{\max}-R_2^{\min}, \cdots, R_n^{\max}-R_n^{\min}\right)$, 是此结点在任一维空间上的最大值($R_i^{\max}, i\in\{1,2,\cdots,n\}$)和最小值($R_i^{\min}, i\in\{1,2,\cdots,n\}$)差值组成的空间向量; R_{dis} 和 $R_{\min}$ 共同将整个结点的数据框住.

3) *parent* 指向 *entity* 的父结点, 其中, 根结点的父结点为空.

4) *childList* 是一个数组: 如果 *entity* 是一个非叶子结点, 那么 $childList[i]$ 指向它的孩子结点; 如果 *entity* 是一个叶子结点, 那么 $childList[i]$ 指向一个数据项 $child=(R_{\min}, R_{dis}, parent, content)$, 其中 $R_{\min}$ 是 1 节中式(1)中数据标准化后的 *behave* 组成的向量(也是空间中一个点), $R_{dis}=(0,0,\cdots,0)$, *content* 是式(1)中的 *url*,*title* 等信息, 用于推送用户时的内容. 图 1(A)简单表示了数据存储的逻辑结构.

算法 1. Rtree_Create

输入: n 个用户兴趣行为向量 $be=\{b_1,b_2,\cdots,b_n\}$

输出: 用户兴趣行为 R 树结构

(1) 将用户兴趣行为 *be* 标准处理, 变换为 z;

(2) 计算与坐标轴相似度:

$$new=\left\{\frac{z_1}{d},\cdots,\frac{z_n}{d}\right\}, d=\sqrt{z_1^2+,\cdots,+z_n^2};$$

(3) 从根结点开始读取 R 树中结点 e;

(4) 如果 e 是非叶子结点, 依次分别比较 e 的所有孩子结点加入 *new* 后扩张空间大小 $\exp ansion$ (由式(3)计算), 选取扩张空间最小的孩子结点 e_{child} 继续执行(4); 如果 e 是叶子结点, 执行(5).

$$\exp ansion = area_{end} - area_{original} \quad (3)$$

其中, $area_{original}=\prod_1^n e^{R_i^{dis}}, R_i^{dis}=R_i^{\max}-R_i^{\min}$,

$area_{end}=\prod_i^n\left(e^{R_i^{dis}}+R_i^{delta}\right)$,

$$R_i^{delta}=\begin{cases} new^{R_i^{\min}+R_i^{dis}}-e^{R_i^{\min}+R_i^{dis}} & if\ \ e^{R_i^{\min}+R_i^{dis}}\le new^{R_i^{\min}+R_i^{dis}} \\ e^{R_i^{\min}}-new^{R_i^{\min}} & if\ \ e^{R_i^{\min}+R_i^{dis}}> new^{R_i^{\min}+R_i^{dis}} \end{cases}.$$

(5) 找到待插入的叶子结点 e_{leaf}, 插入新增结点 e, e_{leaf} 即为 e 的父结点 e_{parent}, 执行(6).

(6) 如果 e_{parent} 孩子结点的个数小于最大值, 则调整结束; 如果 e_{parent} 的孩子结点个数大于最大值, 则分裂 e_{parent} 为两个结点, 继续(6)调整 e_{parent} 的父结点.

3.2 R_CF 算法

算法 2. R_CF

输入: 1 个用户兴趣行为向量 $be=\{b_1,\cdots,b_n\}$

输出: m 个用户的兴趣行为信息

1) 对用户兴趣行为 *be* 进行标准化处理, 变换为 $z=\{z_1,\cdots,z_n\}$;

2) 计算与坐标轴相似度: $goal=\left\{\frac{z_1}{d},\cdots,\frac{z_n}{d}\right\}, d=\sqrt{z_1^2+,\cdots,+z_n^2}$;

3) 从根结点开始读取 R 树中结点 e;

4) 如果 e 是非叶子结点, 依次遍历 e 的所有孩子结点 e_{child}, 执行(5); 如果 e 是叶子结点, 将 e 中的数据插入 top 集中;

5) 如果 *goal* 在任意维度上都在 e_{child} 内(即 $e_{child}^{R_i^{\min}}\le goal_i\le e_{child}^{R_i^{\min}+R_i^{dis}}$), 则选取 e_{child} 执行(4), 否则, 则记录结点 e 及所在树中的层数 *level*.

6) 依次遍历 *level* 最小的结点 e, 将包含的叶子结点的数据插入 top 集中, 对 top 集数据排序, 推荐前 m 个用户.

协同过滤算法需要计算两个用户之间的相似度(即两个点之间的距离), 一个用户兴趣行为即为 n 维空间中一个点, 每个用户的一个兴趣行为即为一个维度. 传统的协同过滤算法常用的两个距离表示方式有欧氏距离和夹角余弦距离. 下面解析一下两个距离在 R 树构造上的不同.

欧氏距离(Euclidean Distance): 一个通常采用的距离定义, 指在 m 维空间中两个点之间的真实距离, 或者向量的自然长度(即该点到原点的距离). 在二维和三维空间中的欧氏距离就是两点之间的实际距离.

两个向量 $A=(a_1,a_2,\cdots,a_n)$ 和 $B=(b_1,b_2,\cdots,b_n)$ 之间的距离 $d(A,B)=\sqrt{\left[\sum_{i=1}^{n}(a_i-b_i)^2\right]}$，其中 $(i=1,2,\cdots,n)$．使用欧氏距离比较标准化后的数据，可以获取 A 用户和 B 用户之间的相似度，距离越小，表示两个用户的相似性越大[8,9].

图 1 是 a、b、c、d、e、f 和 g 七个用户兴趣行为历史数据，图 1(B)是其空间坐标结构，图 1(A)是树型结构表示，用户行为之间的相似度是以欧几里德距离表示．图 1(B)中 *goal* 是需要推送内容的目标用户，通过图 1 中(B)可以看出 *goal* 位于 *cluster*1 结点中，*goal* 和 b、c、d 用户的兴趣行为相似度高于 *cluster*2 中的结点，所以，在只需再计算目标用户 *goal* 和 b、c、d 之间的相似度即可实现 top-n 推荐，在大数据量的情况下，大大节省了目标用户和历史用户数据的比较时间，提高推荐速度．

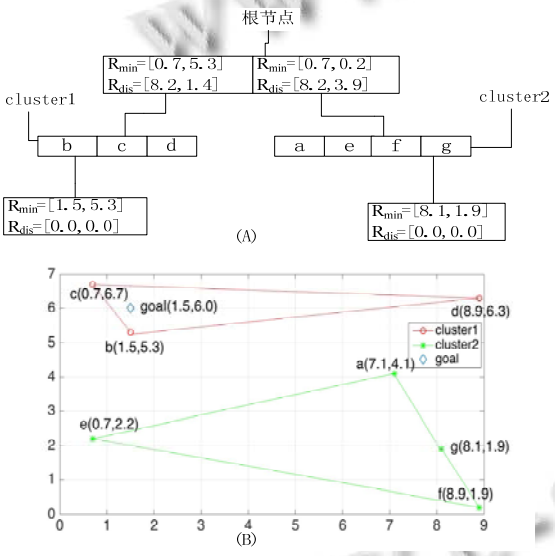

图 1　欧几里德距离表示的相似度

夹角余弦(Cosine)：几何中夹角余弦可用来衡量两个向量方向的差异，机器学习中借用这一概念来衡量样本向量之间的差异．两个 n 维样本点 $a(x11,x12,\cdots,x1n)$ 和 $b(x21,x22,\cdots,x2n)$，可以使用类似于夹角余弦的概念来衡量它们间的相似程度．则 a 和 b 的余弦距离表示为

$$\cos\theta=\frac{\sum_{k=1}^{n}x_{1k}x_{2k}}{\sqrt{\sum_{k=1}^{n}x_{1k}^2}\sqrt{\sum_{k=1}^{n}x_{2k}^2}}$$

其中，本文所获取的用户数据都是正数，所以夹角余弦取值范围为[0,1]．夹角余弦越大表示两个向量的夹角越小，则 a 和 b 两个用户的相似度越高；夹角余弦越小表示两向量的夹角越大，则 a 和 b 两个用户的相似度越低[11-13].

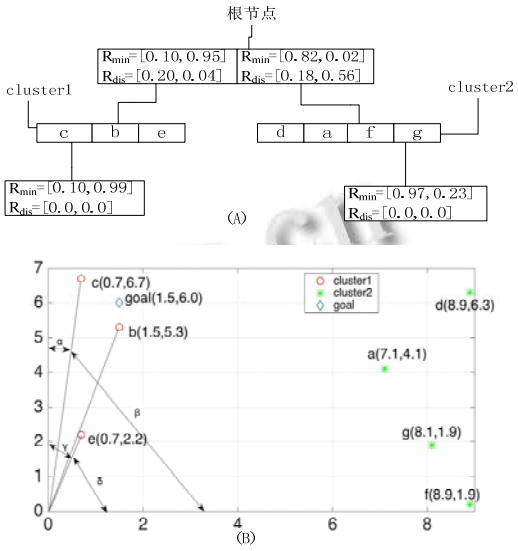

图 2　余弦距离表示的相似度

图 1 是 a、b、c、d、e、f、g 和 *goal* 以欧几里德距离表示的相似度来构造的 R 树，图 2 则是以余弦距离表示的相似度来构造 R 树．图 2 与图 1 同样也是 a、b、c、d、e、f、g 七个用户兴趣行为历史数据，*goal* 是表示需要推送内容的目标用户．图 2(B)可以看出，b、c、e 三个点与横坐标轴形成的余弦距离最小值和最大值分别为

$$\cos\beta=\frac{0.7}{\sqrt{0.7^2+6.7^2}}=0.10\text{、}\cos\delta=\frac{0.7}{\sqrt{0.7^2+2.2^2}}=0.30,$$

与纵坐标轴形成的余弦距离最小值和最大值分别是

$$\cos\gamma=\frac{2.2}{\sqrt{0.7^2+2.2^2}}=0.95\text{、}\cos\alpha=\frac{6.7}{\sqrt{0.7^2+6.7^2}}=0.99.$$

所以，b、c、e 三点隶属于 *cluster*1 结点的 $R_{\min}=[0.10,0.95]$、$R_{dis}=[0.20,0.04]$.

通过算法 1 构造 R 树如图 2，b、c、e 属于一个叶子结点，a、d、g、f 属于一个叶子结点．由图 2(B)可明显看出，*goal* 位于 *cluster*1 结点中，所以，通过算法 2 查询与 *goal* 相似的用户时，无须计算与 a、d、g、f 的相似度，可以优化检索效率．

4　实验分析

为了验证本文提出的基于 R 树的协同过滤推荐算法的可行性和优越性，在 OS X Yosemite 10.10.5 系统

(硬件环境: 8G 内存, 2.7GHz 处理器)下, 采用 java 语言 (JDK 1.8) 和 Eclipse 开发工具实现了传统的协同过滤算法(CF)和基于 R 树的协同过滤算法(R_CF). 相较于传统的协同过滤算法, R_CF 极大的提高算法时间性能, 同时保证了算法的推荐的准确性.

4.1 实验结果分析

图 3 是目标用户分别与 10000、20000、30000、40000、50000、60000、70000、80000、90000、100000 个历史用户兴趣行为信息比较获得相近的 top-100 个用户数两种算法(CF 和 R_CF)的时间性能比较. 从图 3 中可以看出, CF 算法随着用户数量的增加, 耗费的时间逐渐增大, 且增幅明显; 而 R_CF 算法耗费的时间远远低于 CF 算法的运行的时间, 且随着用户数量的增大, 变化的幅度很小.

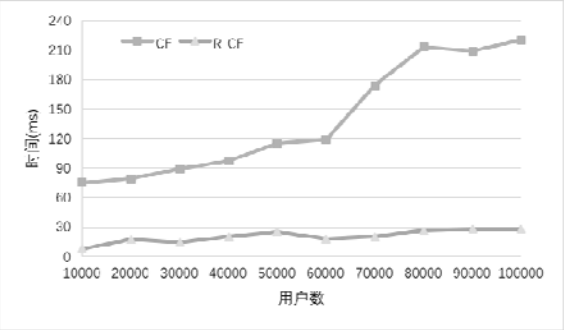

图 3 CF 和 R_CF 算法时间性能比较

表 2 R_CF 算法 top-100 准确率

用户数	准确率(%)
10000	96
20000	95
30000	99
40000	97
50000	96
60000	97
70000	99
80000	99
90000	95
100000	96

表 2 是 R_CF 算法和 CF 算法在用户数为 10000、20000、…、100000 用户数时, 计算的与目标用户相似度排名前一百的相同数目, 也就是 R_CF 算法推荐的 top-100 与 CF 算法计算获得的 top-100 的相同的数目, 可以看出, 与传统的协同过滤算法比较, 推荐正确数都已保持在 95%以上.

5 结语

本文首先从用户隐式反馈的行为信息着手, 构建用户兴趣行为模型. 然后, 通过用户兴趣行为之间的相似度来进行构建 R 树结构, 以用户兴趣行为的相似度作为索引查询 top-n 个相似用户, 并进行实验验证基于 R 树的协同过滤算法极大地提高了查询效率, 从而解决了大数据时代传统协同过滤算法的性能低下的问题.

参考文献

1 印鉴,王智圣,李琪,等.基于大规模隐式反馈的个性化推荐. 软件学报,2014(9):1953–1966.
2 王微微,夏秀峰,李晓明.一种基于用户行为的兴趣度模型. 计算机工程与应用,2012,48(8):148–151.
3 Wang Z, Tu L, Guo Z, et al. Analysis of user behaviors by mining large network data sets. Future Generation Computer Systems, 2014, 37(7): 429–437.
4 Wang SC, Xu GL, Du RJ. Restricted Bayesian classification networks. Science China, 2013, 56(7): 1–15.
5 Lee KCK, Zheng B, Li H, et al. Approaching the Skyline in Z Order. Vldb, 2007: 279–290.
6 Skopal T, Lokoc J. Answering Metric Skyline Queries by PM-tree. Dateso, 2010: 22–37.
7 Lee J, Hwang SW. BSkyTree: scalable skyline computation using a balanced pivot selection. International Conference on Extending Database Technology. ACM. 2010. 195−206.
8 张学胜.面向数据稀疏的协同过滤推荐算法研究[硕士学位论文].合肥:中国科学技术大学,2011.
9 王均波.协同过滤推荐算法及其改进研究[硕士学位论文]. 重庆:重庆大学,2010.
10 Breese JS, Heckerman D, Kadie C. Empirical analysis of predictive algorithms for collaborative filtering. Fourteenth Conference on Uncertainty in Artificial Intelligence, 1998, 7(7): 43–52.
11 Koren Y. Collaborative filtering with temporal dynamics. Communications of the ACM, 2010, 53(4): 89–97.
12 王惠敏.融合用户和项目相关信息的协同过滤算法研究. 武汉理工大学学报,2007,(7):160–163.
13 Tsai CF, Hung C. Cluster ensembles in collaborative filtering recommendation. Applied Soft Computing, 2012, 12(4): 1417–1425.