基于项目聚类的协同过滤推荐算法

邓爱林[1,2], 左子叶[1], 朱扬勇[1]

[1](复旦大学 计算机与信息技术系,数据库中心 上海, 200433)
[2](上海电信技术研究院 上海, 200122)

摘 要: 推荐系统是电子商务中最重要的技术之一,协同过滤是推荐系统中采用最为广泛也是最成功的推荐技术.随着电子商务系统用户数目和商品数目日益增加,在整个用户空间上寻找目标用户的最近邻居非常耗时,导致推荐系统的实时性要求难以保证.针对上述问题,本文提出了一种基于项目聚类的协同过滤推荐算法,根据用户对项目评分的相似性对项目进行聚类,生成相应的聚类中心,在此基础上计算目标项目与聚类中心的相似性,从而只需要在与目标项目最相似的若干个聚类中就能寻找到目标项目的大部分最近邻居并产生推荐列表.实验结果表明,本算法可以有效提高推荐系统的实时响应速度.

关 键 词: 电子商务;推荐系统;协同过滤;聚类;平均绝对偏差
中图分类号: TP311 文献标识码: A 文 章 编 号: 1000-1220(2004)09-1665-06

Collaborative Filtering Recommendation Algorithm Based on Item Clustering

DENG Ai-lin[1,2], ZUO Zi-ye[1], ZHU Yang-yong[1]

[1](*Computer and Information Technology Department Fudan University, Shanghai* 200433, *China*)
[2](*Shanghai Academy of Telecom Technology, Shanghai* 200122, *China*)

Abstract: Recommendation system is one of the most important techniques used in E-Commerce. Many recommendation systems employ collaborative filtering to generate recommendations. With the gradual increase of users and commodities in E-Commerce, the time-consuming nearest neighbor search of the target user in the total user space resulted in the failure of ensuring the real-time requirement of recommendation system. A collaborative filtering recommendation algorithm based on item clustering was proposed in this paper to solve this problem. Items were clustered based on users' ratings on items, each cluster has a cluster center. Based on the similarity between target item and cluster centers, the nearest neighbors of target item can be found in the item clusters that most similar to the target item. Experimental results indicated that this algorithm could effectively improve the real-time performance of recommendation systems.

Key words: E-Commerce; recommendation systems; collaborative filtering; clustering; MAE

1 引 言

随着互联网的普及和电子商务的发展,推荐系统逐渐成为电子商务IT技术的一个重要研究内容,得到越了来越多研究者的关注[1].目前,几乎所有大型的电子商务系统,如Amazon、CDNOW、eBay、当当网上书店等,都不同程度的使用了各种形式的推荐系统.

最近邻协同过滤推荐是当前最成功的推荐技术[2],其基本思想就是基于评分相似的最近邻居的评分数据向目标用户产生推荐.由于最近邻居对项目(电影,音乐等)的评分与目标用户对该项目的评分非常相似,因此目标用户对未评分项目的评分可以通过最近邻居对该项目评分的加权平均值逼近.

最近邻协同过滤推荐需要在整个用户空间上搜索目标用户的最近邻居,随着电子商务系统规模越来越大,用户数量和项目数量急剧增加,在整个用户空间上搜索目标用户的最近邻居非常耗时,越来越难以满足推荐系统的实时性要求[3].针对电子商务推荐系统存在的上述问题,本文提出了一种基于项目聚类的协同过滤推荐算法,通过用户对项目评分的相似性对项目进行聚类,将用户评分比较类似的商品加入同一个聚类中,然后根据每个聚类中用户对商品的评分生成对应的聚类中心,在此基础上计算目标项目与聚类中心的相似性,选择与目标项目相似性最高的若干个聚类作为查询空间,在这些聚类中搜索目标项目的最近邻居,从而能够在尽量少的项目空间上搜索到目标项目尽可能多的最近邻居,最后根据用户对最近邻居的评分预测该用户对目标项目的评分并产生最终的推荐列表.对项目进行聚类比较耗时,但可以离线进行.实验结果表明,本文提出的方法能有效提高推荐系统的实时响应速度.

收稿日期: 2003-03-04 收修改稿日期: 2003-07-05 基金项目: 国家863计划(2001AA113181)资助 作者简介: 邓爱林,博士研究生,主要研究领域为数据库与知识库. E-mail: dal5758@sina.com; 左子叶,硕士研究生,主要研究领域为数据库与知识库; 朱扬勇,博士,教授,博士生导师,主要研究领域为数据库与知识库.

2 相关工作

为了产生精确而有效的推荐,保证推荐系统的实时性要求,研究者提出了各种不同的推荐算法,如协同过滤推荐、基于项目的协同过滤推荐、Bayesian 网络技术、聚类技术、关联规则技术以及基于图的 Horting 图技术等.

Tapestry[4] 是最早提出的协同过滤推荐系统,目标用户需要明确指出与自己行为比较类似的其他用户. GroupLens[5] 是基于用户评分的自动化协同过滤推荐系统,用于推荐电影和新闻. Ringo 推荐系统[6] 和 Video 推荐系统[7] 通过电子邮件的方式推荐音乐和电影. Breese 等人[2] 对各种协同过滤推荐算法及其改进进行了深入分析.

传统的协同过滤推荐根据用户的最近邻居产生最终的推荐,基于项目的协同过滤推荐首先计算项目之间的相关性,然后通过用户对相关项目的评分预测用户对未评分项目的评分[8].

Bayesian 网络技术利用训练集创建相应的模型[9],模型用决策树表示,节点和边表示用户信息. 训练得到的模型非常小,所以对模型的应用非常快. 这种方法适合用户的兴趣爱好变化比较慢的场合.

聚类技术将具有相似兴趣爱好的用户分配到相同的聚类中[10,11],聚类产生之后,根据聚类中其他用户对商品的评价预测目标用户对该商品的评价. 这种方法存在的最大缺陷在于如果目标用户处于聚类的边缘,则对该用户的推荐精度比较低. Mobasher. B 等人[12] 提出通过对服务器日志进行事务聚类和关联规则超图分割聚类(Association Rule Hypergraph Partitioning, ARHP)获取用户的共同浏览特征,然后扫描所有的数据集产生个性化的推荐. O'Connor. M 等人[13] 提出对项目进行聚类,然后在对应的聚类中搜索目标用户的最近邻居,由于每个聚类中的用户数量并不是随着聚类中项目数量的减少而线性减少,所以这种方法在用户对多个聚类中的商品均有评分的情况下效果并不理想.

关联规则技术在零售业得到了广泛的应用,关联规则挖掘可以发现不同商品在销售过程中的相关性. 基于关联规则的推荐算法与协同过滤推荐算法不同,协同过滤推荐算法根据用户评分数据产生推荐,而基于关联规则的推荐算法根据用户交易数据生成关联规则模型,应用该模型和用户当前的购买行为向用户产生推荐[14]. 关联规则模型的生成可以离线进行.

Horting 图技术是一种基于图的方法[15],节点代表用户,边代表两个用户之间的相似度. 在图中搜索近邻节点,然后综合近邻节点的评分形成最后的推荐. Horting 图技术可以跳过中间节点寻找最近邻居,考虑了节点之间的传递相似关系. 因此推荐精度优于最近邻协同过滤技术.

针对用户评分数据的极端稀疏性,文[16]提出通过奇异值分解(SVD)减少项目空间的维数,使得用户在减少的项目空间上对每一个项目均有评分,实验结果表明这种方法可以有效解决同义词(Synonymy)问题,显著提高推荐系统的伸缩能力. 但降维会导致信息损失,降维效果与数据集密切相关,在项目空间维数很高的情况下,降维的效果难以保证[17].

3 基于项目聚类的协同过滤推荐算法

基于项目的协同过滤推荐根据用户对相似项目的评分预测该用户对目标项目的评分[8],它基于这样一个假设:如果大部分用户对一些项目的评分比较相似,则当前用户对这些项目的评分也比较相似. 基于项目的协同过滤推荐系统使用统计技术找到目标项目的若干最近邻居,由于当前用户对最近邻居的评分与对目标项目的评分比较类似,所以可以根据当前用户对最近邻居的评分预测当前用户对目标项目的评分,产生对应的推荐列表.

随着电子商务系统规模的扩大,用户数目和项目数目指数级增长,在大型电子商务系统中,用户评分的项目一般不会超过项目总数的 1%[8],基于项目的协同过滤推荐需要在整个项目空间上查询目标项目的最近邻居,这对推荐算法的伸缩能力是一个极大的挑战,推荐系统的实时性要求越来越难以满足.

基于项目聚类的协同过滤推荐算法通过用户对项目评分的相似性对项目进行聚类并生成对应的聚类中心,在此基础上计算目标项目与聚类中心的相似性,选择与目标项目相似性最高的若干个聚类作为查询空间,在这些聚类中搜索目标项目的最近邻居,从而能够在尽量少的项目空间上搜索到目标项目大部分的最近邻居,有效提高推荐系统的实时响应速度.

在对项目进行聚类和搜索目标项目最近邻居的过程中,都需要度量项目之间的相似性. 用户评分数据可以用一个 m^*n 阶矩阵 $A(m,n)$ 表示,m 行代表 m 个用户,n 列代表 n 个项目,第 i 行第 j 列的元素 $R_{i,j}$ 代表用户 i 对项目 j 的评分. 用户评分数据矩阵如图 1 所示.

	$Item_1$	…	$Item_k$	…	$Item_n$
$User_1$	$R_{1,1}$	…	$R_{1,k}$	…	/
…	…	…	…	…	…
$User_j$	$R_{j,1}$	…	/	…	$R_{j,n}$
…	…	…	…	…	…
$User_m$	/	…	$R_{m,k}$	…	$R_{m,n}$

图 1 用户评分数据矩阵

Fig. 1 User rating data matrix

度量项目 i 和项目 j 之间相似性的方法如下,首先得到对项目 i 和项目 j 评分过的所有用户,然后通过不同的相似性度量方法计算项目 i 和项目 j 之间的相似性,记为 $sim(i,j)$.

度量项目间相似性的方法有许多种,主要包括如下三种方法[8]:余弦相似性、相关相似性以及修正的余弦相似性.

· 余弦相似性:项目评分看作为 m 维用户空间上的向量,如果用户对项目没有进行评分,则将用户对该项目的评分设为 0,项目间的相似性通过向量间的余弦夹角度量. 设项目 i 和项目 j 在 m 维用户空间上的评分分别表示为向量 $\vec{i}$,$\vec{j}$,则项目 i 和项目 j 之间的相似性 $sim(i,j)$ 为:

$$sim(i,j)=\cos(\vec{i},\vec{j})=\frac{\vec{i}\cdot\vec{j}}{\|\vec{i}\|*\|\vec{j}\|}$$

分子为两个项目评分向量的内积,分母为两个项目评分向量模的乘积.

· 相关相似性:设对项目i和项目j共同评分过的用户集合用U_{ij}表示,则项目i和项目j之间的相似性$sim(i,j)$通过Pearson相关系数度量:

$$sim(i,j)=\frac{\sum_{c\in U_{ij}}(R_{c,i}-\overline{R_i})(R_{c,j}-\overline{R_j})}{\sum_{c\in U_{ij}}(R_{c,i}-R_i)^2\ \overline{\sum_{c\in U_{ij}}(R_{c,j}-\overline{R_j})^2}}$$

$R_{c,i}$表示用户c对项目i的评分,$\overline{R_i}$和$\overline{R_j}$分别表示对项目i和项目j的平均评分.

· 修正的余弦相似性:在余弦相似性度量方法中没有考虑不同用户的评分尺度问题,修正的余弦相似性度量方法通过减去用户对项目的平均评分改善上述缺陷,设对项目i和项目j共同评分过的用户集合用U_{ij}表示,U_i和U_j分别表示对项目i和项目j评分过的用户集合,则项目i和项目j之间的相似性sim(i, j)为:

$$\mathrm{sim}(i,j)=\frac{\sum_{c\in U_{ij}}(R_{c,i}-\overline{R_c})(R_{c,j}-R_c)}{\overline{\sum_{c\in U_i}(R_{c,i}-\overline{R_c})^2}\ \overline{\sum_{c\in U_j}(R_{c,j}-\overline{R_c})^2}}$$

$R_{c,i}$表示用户c对项目i的评分,$\overline{R_c}$表示用户c对项目的平均评分.

3.1　项目聚类

根据上述的项目相似性度量方法,可以对项目进行聚类.所谓聚类,就是将数据对象分组成多个类(聚类),从而使得同一个聚类中对象之间具有较高的相似性,而不同聚类中的对象差别较大[18].

下面介绍通过K-Means聚类算法对项目进行聚类的具体算法,简称为ItemClusteringByKMeans.算法的具体步骤如下:

算法**1**. ItemClusteringByKMeans(s, $URDB$)
输入:聚类数目s和用户评分数据库$URDB$
输出:s个聚类
方法:

1) 从用户评分数据库$URDB$中检索所有n个项目,记为集合$I=\{i_1,i_2,\cdots,i_n\}$;
2) 从用户评分数据库$URDB$中检索所有m个用户,记为集合$U=\{u_1,u_2,\cdots,u_m\}$;
3) 任意选择s个项目,将用户评分数据库$URDB$对这s个项目的评分作为初始的聚类中心,记为集合$CC=\{cc_1,cc_2,\cdots,cc_s\}$;
4) s个聚类$c_1,c_2,\cdots,c_s$均初始化为空,记为集合$C=\{c_1,c_2,\cdots,c_s\}$;
5) repeat
　for each item $i_i\in I$
　　for each cluster center $cc_j\in CC$
　　计算项目i_i和聚类中心cc_j的相似性$sim(i_i,cc_j)$;
　　endfor
　　$sim(i_i,cc_m)=\max\{sim(i_i,cc_1),sim(i_i,cc_2),\cdots,sim(i_i,cc_s)\}$;
　　聚类$c_m=c_m\cup i_i$;
　endfor
　for each cluster $c_i\in C$
　　for each user $u_j\in U$
　　计算用户u_j对聚类c_i中所有项目的平均评分$R_{uj,ci}$,生成新的聚类中心cc_i;
　　endfor
　endfor
6) until 聚类$c_1,c_2,\cdots,c_s$上一轮循环中的聚类$c_{1old},c_{2old},\cdots,c_{sold}$相同
7) 返回;

通过ItemClusteringByKMeans项目聚类算法可以将用户评分比较相似的项目分配到同一个聚类中,从而使得用户对同一个聚类中项目评分的相似性尽可能高,而对不同聚类间项目评分的相似性尽可能低.设整个项目空间用集合$I=\{i_1,i_2,,i_n\}$表示,生成的s个聚类用集合$C=\{c_1,\cdots,c_s\}$表示,则用户对每个聚类$c_i(i=1,\cdots,s)$中包含的项目评分尽可能相似.并有如下性质:

· $c_1\cup c_2\cup\cdots\cup c_s=I$

· $c_i\cap c_j=\Phi(i\neq j$,对任意的$1\le i\le s,\ 1\le j\le s)$

生成的s个聚类对应s个聚类中心,用集合$CC=\{cc_1,\cdots,cc_s\}$表示,每个聚类中心$cc_i(i=1,\cdots,s)$表示用户对该聚类$c_i(i=1,\cdots,s)$中项目的平均评分.聚类中心作为该聚类的代表,表示用户对该聚类中项目的典型评分.

根据生成的s个聚类中心构造对应的聚类中心评分数据矩阵$ClusteCenterRatingMatrix(m,s)$,聚类中心评分数据矩阵如图2所示,$m$行代表$m$个用户,$s$列代表$s$个聚类中心,第$i$行第$j$列的元素$R_{i,j}$代表用户$i$对项目聚类$j$中项目的平均评分.

	Cc_1	…	cc_i	…	cc_s
$User_1$	$R_{1,1}$	…	$R_{1,i}$	…	/
…	…	…	…	…	…
$User_j$	$R_{j,1}$	…	/	…	$R_{j,s}$
…	…	…	…	…	…
$User_m$	/	…	$R_{m,i}$	…	$R_{m,s}$

图2　聚类中心评分数据矩阵

Fig. 2　Item cluster center rating matrix

聚类中心评分数据矩阵$ClusteCenterRatingMatrix(m,s)$用于计算目标项目与聚类中心的相似性,从而选择与目标项目相似性最高的前若干个聚类作为搜索空间,在此搜索空间中搜索目标项目的最近邻居.对项目进行聚类比较费时,但可以离线周期进行.

3.2　基于项目聚类的最近邻查询

由聚类的性质可知,目标项目的最近邻居大部分分布在与目标项目相似性最高的若干个聚类中,因此不需要在整个项目空间上查询目标项目的最近邻居,而只需要在与目标项目相似性最高的若干聚类中就能查询到目标项目的大部分邻居.由于与目标项目相似性最高的若干聚类想对于整个项目空间项目而言,其搜索空间要小得多,因此本文提出的方法可以大大提高在线的最近邻查询速度,有效满足推荐系统的实

时性要求.

下面详细介绍基于项目聚类的最近邻查询算法,简称为 KNNByItemClustering.

算法**2**. KNNByItemClustering (TI, k, $URDB$, C, CC, ϵ)

输入:目标项目TI,目标项目的最近邻居数目k,用户评分数据库$URDB$,项目聚类C,项目聚类中心CC 和相似性阈值 ϵ

输出:目标项目的k 个最近邻居

方法:

```
1) CanItemSet = Φ;
2) For (int i = 1, i < = s, i+ + )
3)     If(sim(cc_i, TI) ≥ ε)
4)         CanItemSet = CanItemSet + c_i;
5)     endif
6) endfor
7) for each item ∈ CanItemSet
8)     计算item 与TI 之间的相似性sim(TI, item);
9) endfor
10) 根据 CanItemSet 中项目与目标项目TI 之间的相似性,
选择相似性最高的前k 个项目作为目标项目的最近邻居;
11) 返回;
```

算法第2步到第6步计算目标项目TI 与所有聚类中心的相似性,如果目标项目TI 与某聚类中心的相似性大于事先指定的相似性阈值 ϵ,则将相应聚类中的所有项目加入到候选项目集 $CanItemSet$ 中. 算法第 7 步到第 9 步在项目空间 $CanItemSet$ 中查找目标项目 TI 的最近邻居集合 $NN = \{NN_1, NN_2, \ldots, NN_k\}$,使得$TI \notin NN$,并且$NN_1$ 与目标项目TI 的相似性$sim(TI, NN_1)$ 最高,NN_2 与目标项目TI 的相似性$sim(TI, NN_2)$次之,依此类推.

算法分析:由于聚类中心集合远小于整个项目空间集合,即 $\| CC \| << \| I \|$,因此计算目标项目TI 与所有聚类中心相似性的时间代价可以忽略不计. 在进行目标项目的最近邻查询时,只需要在与目标项目相似性最高的若干个聚类组成的候选项目集$CanItemSet$ 中而不是整个项目空间P 中查询就能找到目标项目的大部分邻居,由于候选项目集远小于整个项目空间集合,即 $\| CanItemSet \| << \| I \|$,因此本文提出的方法可以有效提高推荐系统的最近邻查询速度.

基于项目聚类的最近邻查询只在与目标项目相似性最高的若干聚类中查询到目标项目的最近邻居,因此只能保证搜索到目标项目的大部分邻居,但并不能保证搜索到目标项目的所有最近邻居,所以可能会降低推荐系统的推荐精度.

基于项目聚类的最近邻查询的相似性阈值(可以自动进行调整,当推荐系统探测到推荐精度降低的时候(如大部分推荐用户不感兴趣),可以适当降低相似性阈值(,从而在更大的项目空间上查询目标项目的最近邻居以提高推荐系统的推荐精度,但会降低推荐系统的实时响应速度.当推荐系统探测到响应时间比较长时,可以适当增加相似性阈值(,从而在更小的项目空间上查询目标项目的最近邻居以提高推荐系统的实时响应速度.

3.3 产生推荐

通过本文提出的方法得到目标项目的最近邻居后,下一步需要产生相应的推荐. 设目标项目 TI 的最近邻居集合用 $NN_{TI} = \{NN_1, NN_2, \ldots, NN_k\}$ 表示,则用户 u 对项目 TI 的预测评分P_{uTI}可以通过用户u 对最近邻居集合NN_{TI}中项目的评分得到,计算方法如下[2]:

$$P_{u,TI} = \overline{R_{TI}} + \frac{\sum_{n \in NN_{TI}} sim(TI, n) * (R_{u,n} - R_n)}{\sum_{n \in NN_{TI}} (|sim(TI, n)|)}$$

$sim(TI, n)$ 表示目标项目 TI 与最近邻居 n 之间的相似性,$R_{r,n}$表示用户 u 对项目 n 的评分. $\overline{R_{TI}}$和$\overline{R_n}$分别表示对项目 TI 和项目 n 的平均评分.

通过上述方法预测当前用户对所有未评分项目的评分,然后选择预测评分最高的前若干个项目作为推荐结果反馈给当前用户.

4 实验结果及其分析

4.1 数据集

本文采用 MovieLens 站点提供的数据集(*http://movielens.umn.edu/*),MovieLens 是一个基于Web 的研究型推荐系统,用于接收用户对电影的评分并提供相应的电影推荐列表. 目前,该Web 站点的用户已经超过 43000 人,用户评分的电影超过3500部.

我们从用户评分数据库中选择20851 条评分数据作为实验数据集,实验数据集中共包含930 个用户和 438 部电影,其中每个用户至少对20部电影进行了评分.

4.2 试验结果及分析

本文将从两个方面进行试验检验本算法的有效性. 首先验证本算法是否能够在尽量少的项目空间上搜索到尽可能多的目标项目的最近邻居,然后检验本算法对推荐系统推荐精度的影响.

在3.1 节中介绍的三种项目间相似性度量方法中,余弦相似性度量方法简单明了,可以有效度量项目间的相似性,而且计算速度比较快,因此本文在对项目进行聚类和进行目标项目的最近邻查询时均采用余弦相似性度量方法.

4.2.1 最近邻查询效率试验

为了检验本算法的有效性,设目标项目为 TI,整个项目空间用I 表示,首先在整个项目空间上作最近邻查询,本文中选择最近邻居数目为10,查询结果记为集合NN_o;然后在与目标项目TI 最相似的前k 个聚类(记为$c_1, c_2, \cdots, c_k$)中作最近邻查询,最近邻居数目也选择为 10,查询结果记为集合 NN_k,则本算法的有效性可以表述为只需要扫描原始数据集的($\| c_1 \| + \| c_2 \| + \cdots + \| c_k \|$) / $\| I \|$ (百分比) 就能找到目标项目TI 在全部项目空间上 $\| NN_k \cap NN_o \|$ / $\| NN_o \|$ (百分比) 的邻居. 为了达到统计上的显著性,本文将在整个项目空间上统计上述指标.

在对项目进行聚类过程中,聚类数目的指定非常关键,聚类数目指定过大,则计算目标项目与聚类中心的相似性需要耗费大量时间,不能有效提高推荐系统的实时响应速度;聚类

数目指定过小,则每个聚类中包含的项目比较多,即使在与目标项目最相似的若干聚类中进行最近邻居查询也需要扫描大量的候选项目集,也不能有效提高推荐系统的实时响应速度.本文分别以10、20和30作为聚类数目对438部电影进行聚类,实验结果分别如图3、图4和图5所示.

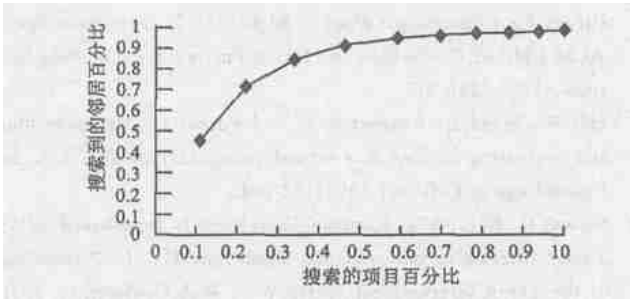

图3　聚类数目为10时最近邻查询效率

Fig.3　Cluster Number = 10

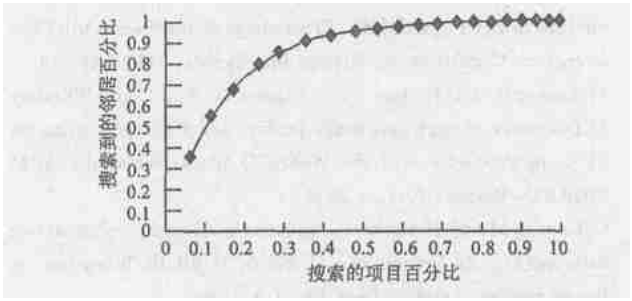

图4　聚类数目为20时最近邻查询效率

Fig.4　Cluster Number = 20

实验结果显示,在指定最近邻查询的最近邻居数目为10的情况下,当聚类数目为10时,只需要扫描整个项目集的34%就可以找到整个项目集上85%的最近邻居,只需要扫描整个项目集的46%就可以找到整个项目集上92%的最近邻居;当聚类数目为20时,只需要扫描整个项目集的28%就可以找到整个项目集上84%的最近邻居,只需要扫描整个项目集的40%就可以找到整个项目集上93%的最近邻居;当聚类数目为30时,只需要扫描整个项目集的23%就可以找到整个项目集上85%的最近邻居,只需要扫描整个项目集的35%就可以找到整个项目集上92%的最近邻居.实验结果表明,本文提出的基于项目聚类的协同过滤推荐算法能够保证在尽量小的项目空间上查询到目标项目尽量多的最近邻居,从而有效提高推荐系统的实时响应速度.

从实验结果中可以发现,聚类数目越大,查找目标项目的最近邻居越快,不过上述结论是在聚类数目远小于项目数目,计算目标项目与聚类中心相似性的时间代价相对于最近邻查询可以忽略不计的条件下才成立,当聚类数目很大的时候,计算目标项目与聚类中心相似性的代价并不能忽略不计.如果指定聚类数目等于项目数目,则本文提出的最近邻查询方法与在全部空间上查询最近邻的时间代价一致.

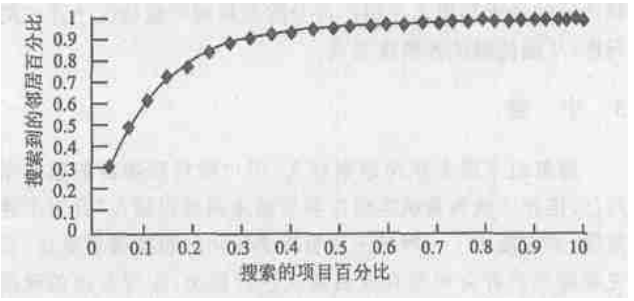

图5　聚类数目为30时最近邻查询效率

Fig.5　Cluster Number = 30

4.2.2　推荐精度试验

上述实验结果表明本文提出的方法并不能保证可以找到整个项目空间上全部的最近邻居.因为目标项目的某些最近邻居可能分布在与目标项目相似性并不高的项目聚类中.下面对本文提出的算法与Sarwar, B.等人[8]提出的在全部项目空间上查询目标项目最近邻居的Item-based推荐算法进行比较,检验本文提出的方法对推荐系统推荐精度的影响.

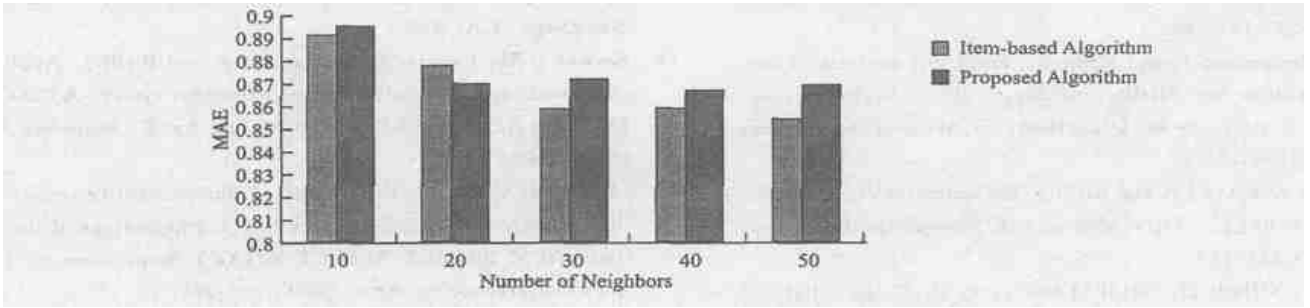

图6　推荐算法推荐精度比较

Fig.6　Comparison of the accuracy of different recommendation algorithms

评价推荐系统推荐精度的度量标准主要包括统计精度度量方法和决策支持精度度量方法两类[8].本文采用统计精度度量方法中被广泛采用的平均绝对偏差MAE(Mean

Absolute Error)作为推荐精度度量标准. 平均绝对偏差MAE通过计算预测的用户评分与实际的用户评分之间的偏差度量预测的准确性, MAE越小, 推荐质量越高. 设预测的用户评分集合表示为$\{p_1, p_2, \ldots, p_N\}$, 对应的实际用户评分集合为$\{q_1, q_2, \ldots, q_N\}$, 则平均绝对偏差$MAE$定义为[8]:

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N}$$

试验过程中, 我们指定项目聚类的聚类数目为30, 进行最近邻搜索时只扫描与目标项目最相似的前6个项目聚类. 目标用户的最近邻居个数从10增加到50, 间隔为10, 分别计算本文提出的算法与Sarwar, B.等人[8]提出的Item-based推荐算法的MAE, 试验结果如图6所示.

实验结果表明, 本文提出的算法与*Item-based*推荐算法相比, 其推荐精度并没有显著降低. 在大部分的情况下, 本文提出算法的MAE与传统推荐算法的MAE类似, 在某些特定条件下(如上述实验中当指定目标项目的最近邻居数目为20时), 本文提出的算法具有更高的推荐精度. 原因可能是在用户评分数据极端稀疏的情况下, 在整个项目空间上搜索到的目标项目的最近邻居并不准确[2]. 而在与目标项目相似性最高的若干项目聚类中搜索目标项目的最近邻居可以有效缓解传统的最近邻搜索在用户评分数据极端稀疏情况下存在的问题, 从而使得推荐精度更高.

5 小 结

随着电子商务规模越来越大, 用户数目和项目数据急剧增加, 推荐系统的系统实时性要求越来越难以满足. 针对上述问题, 本文提出了一种基于项目聚类的协同过滤推荐算法, 首先根据用户评分对所有项目离线进行聚类, 生成对应的聚类中心, 然后根据目标项目与聚类中心的相似性, 只需要扫描与目标项目相似性最高的前若干个聚类就可以找到目标项目的大部分最近邻居, 从而有效提高推荐系统的实时响应速度. 实验结果表明, 本文提出的基于项目聚类的协同过滤推荐算法可以显著提高推荐系统的在线响应速度, 从而有效解决推荐系统处理大规模数据面临的实时性问题.

References:

1 Schafer J B, Konstan J A and Riedl J. Recommender systems in E-Commerce[C]. In: ACM Conference on Electronic Commerce (EC99), 1999, 158-166.

2 Breese J, Hecherman D and Kadie C. Empirical analysis of predictive algorithms for collaborative filtering [C]. In: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence(UAI-98), 1998, 43-52.

3 Schafer J B, Konstan J A and Riedl J. E-Commerce recommendation applications[J]. Data Mining and Knowledge Discovery, 2001, 5(1-2): 115-153.

4 Goldberg D, Nichols D, Oki B M and Terry D. Using collaborative filtering to weave an information tapestry[J]. Communications of the ACM, 1992, 35(12): 61-70.

5 Resnick P, Iacovou N, Suchak M, Bergstrom P and Riedl J. Grouplens. an open architecture for collaborative filtering of netnews[C]. In: Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work, 1994, 175-186.

6 Shardanand U and Maes P. Social information filtering: algorithms for automating "Word of Mouth"[C]. In Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems, 1995, 210-217.

7 Hill W, Stead L, Rosenstein M and Furnas G. Recommending and evaluating choices in a virtual community of Use[C]. In: Proceedings of CHI'95, 1995, 194-201.

8 Sarwar B, Karypis G, Konstan J and Riedl J. Item-based collaborative filtering recommendation algorithms [C]. In: Proceedings of the Tenth International World Wide Web Conference, 2001, 285-295.

9 Chickering D and Hecherman D. Efficient approximations for the marginal likelihood of bayesian networks with hidden variables [J]. Machine Learning, 1997, 29, 181-212.

10 Dempster A, Laird N and Rubin D. Maximum likelihood from incomplete data via the EM algorithm[J]. Journal of the Royal Statistical Society, 1977, 38(1): 1-38.

11 Thiesson B, Meek C, Chickering D and Heckerman D. Learning mixture of DAG models[R]. Proceedings of the Fourteenth Conference on Uncertainty in Articial Intelligence, 1998, 504-513.

12 Mobasher B, Dai H, Luo T, Nakagawa N, Sun Y and Wiltshire J. Discovery of aggregate usage profiles for Web personalization [C]. In Proceedings of the WebKDD Workshop at the ACM SIGKKD, Boston, August 2000.

13 O'Conner M and Herlocker J. Clustering items for collaborative filtering[C]. In Proceedings of the ACM SIGIR Workshop on Recommender Systems. Berkeley, CA, 1999.

14 Sarwar B, Karypis G, Konstan J and Riedl J. Analysis of recommendation algorithms for E-commerce [C]. ACM Conference on Electronic Commerce, 2000, 158-167.

15 Wolf J, Aggarwal C, Wu K-L and Yu P. Horting Hatches an Egg. A new graph-theoretic approach to collaborative filtering [C]. Proceedings of the Fifth ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'99), 201-212. San Diego, CA, 1999.

16 Sarwar B M, Karypis G, Konstan J A and Riedl J. Application of dimensionality reduction in recommender system-A Case study [C]. In: ACM WebKDD Web Mining for E-Commerce Workshop, 2000.

17 Aggarwal C C. On the effects of dimensionality reduction on high dimensional similarity search [C]. Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 2001, 256-266.

18 Jiawei H and Micheline K. Data Mining: Concepts and Techniques. Morgan Kaufmann, 2000 [EB/OL]. http://www.cs.sfu.ca/~han/.