一种融合聚类与用户兴趣偏好的协同过滤推荐算法

何 明 孙 望 肖 润 刘伟世
(北京工业大学信息学部计算机学院 北京 100124)

摘 要 协同过滤推荐算法可以根据已知用户的偏好预测其可能感兴趣的项目,是现今最为成功、应用最广泛的推荐技术。然而,传统的协同过滤推荐算法受限于数据稀疏性问题,推荐结果较差。目前的协同过滤推荐算法大多只针对用户-项目评分矩阵进行数据分析,忽视了项目属性特征及用户对项目属性特征的偏好。针对上述问题,提出了一种融合聚类和用户兴趣偏好的协同过滤推荐算法。首先根据用户评分矩阵与项目类型信息,构建用户针对项目类型的用户兴趣偏好矩阵;然后利用K-Means算法对项目集进行聚类,并基于用户兴趣偏好矩阵查找待估值项所对应的近邻用户;在此基础上,通过结合项目相似度的加权Slope One算法在每一个项目类簇中对稀疏矩阵进行填充,以缓解数据稀疏性问题;进而基于用户兴趣偏好矩阵对用户进行聚类;最后,面向填充后的评分矩阵,在每一个用户类簇中使用基于用户的协同过滤算法对项目评分进行预测。实验结果表明,所提算法能够有效缓解原始评分矩阵的稀疏性问题,提升算法的推荐质量。

关键词 推荐系统,协同过滤,聚类算法
中图法分类号 TP391 **文献标识码** A

Collaborative Filtering Recommendation Algorithm Combining Clustering and User Preferences

HE Ming SUN Wang XIAO Run LIU Wei-shi
(College of Computer Science, Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China)

Abstract Collaborative filtering recommendation algorithm can use the known user preferences to predict the possible interest items, and it is now the most successful and widely used recommendation technique. However, traditional collaborative filtering recommendation algorithms suffer from data sparsity, which results in poor recommendation accuracy. Only user-item rating matrix has been used to data analysis by most current collaborative filtering algorithms, while the characteristics of the item attributes and user preferences for those who have not been considered. To address this issue, a collaborative filtering recommendation algorithm combing clustering and user preference was proposed in this paper. Firstly, the user preference matrix for the item category is constructed according to the user rating matrix and the item category information. Then the K-Means algorithm is used to cluster the item set, and the nearest-neighbor user corresponding to the unrated item is found based on the user preference matrix. Next, the sparse matrix in each item cluster is filled by the weight Slope One algorithm combined with the similarity of items to alleviate the problem of data sparsity. In addition, the user clusters are built based on the user interest matrix. Finally, the user based collaborative filtering algorithm in each user cluster is employed to predict the item rating for the filled rating matrix. The experimental results show that our algorithm effectively alleviates the sparsity problem of the raw rating matrix and achieves better quality of recommendation than other algorithms.

Keywords Recommender systems, Collaborative filtering, Clustering algorithm

1 引言

在信息大数据时代,用户的个性化需求不断提高,这对信息系统的智能度提出了很多挑战。面对大量的数据信息,如何帮助用户有效获取其所需要的信息,有力改善“信息过载”问题,是数据科研工作者的主要研究挑战之一。

为了解决“信息过载问题”,并帮助用户快速准确地获得满足其自身需要的信息,推荐系统(Recommender Systems)[1]应运而生。作为一种有效的信息过滤手段,推荐系统是当前解决“信息过载”问题及实现个性化服务的有效方法之一。推荐系统的技术核心是推荐算法,目前比较常用的推荐算法包括基于内容的推荐、协同过滤推荐和混合推荐等。

协同过滤推荐算法是推荐系统中应用最广泛、最成功的推荐技术之一,具有不依赖于项目的特征信息,不受限于内容

本文受国家自然科学基金(91646201,91546111),国家科技支撑计划子课题(2013BAH21B02-01),北京市教委项目(KZ20160005009,KM201710005023)资助。

何 明(1975-),男,博士,副教授,主要研究方向为推荐系统、数据挖掘、机器学习,E-mail:heming@bjut.edu.cn;孙 望(1991-),男,硕士生,主要研究方向为推荐系统、数据挖掘;肖 润(1990-),男,硕士生,主要研究方向为信息检索;刘伟世(1989-),男,硕士生,主要研究方向为机器学习。

分析技术的限制等优点，这使得该技术在理论研究和实际应用中都取得了很大的发展，但鉴于数据稀疏性、冷启动和隐私保护等问题的限制[2-4]，需要对其不断完善。数据稀疏性问题是指用户-项目评分信息稀疏，用户间共同消费的项目很少，直接导致推荐结果的准确度下降。

目前，研究人员提出了许多方法来克服上述问题，以改善算法的推荐结果。吴湖等人[5]基于联合聚类算法和加权非负矩阵分解算法提出了一种两阶段评分预测方法。韦素云等人[6]通过项目聚类计算用户间的局部相似度，然后通过求解用户全局最近邻的方式来提高相似度计算的准确性。Huang 等人[7]提出一种基于项目的协同过滤实时流推荐系统，利用人口统计学聚类算法解决数据稀疏性问题。郭兰杰等[8]通过构建社交网络，选择性填充评分矩阵，以缓解稀疏性问题，提高相似度计算的准确率。

上述研究都在一定程度上缓解了协同过滤算法的数据稀疏性问题，提高了协同推荐算法的准确性及可扩展性。但这些方法的处理策略过度依赖于用户-项目评分矩阵，用户(项目)之间存在的一些内在信息并没有得到充分利用。我们认为，融合用户和项目特征的协同过滤推荐算法可以提高推荐质量。

基于上述原因，本文针对协同过滤推荐算法中的一个经典问题——数据稀疏性问题进行研究。为了能够有效缓解用户-项目矩阵评分信息的稀疏性，本文根据用户所评价项目的类型，构建针对“用户-项目类型”的用户兴趣偏好矩阵，并引入聚类算法，对评分矩阵在项目维度上进行聚类，在每一个项目类簇中通过用户兴趣偏好矩阵查找待估值项所对应用户的近邻用户；然后应用本文改进的 Slope One 算法对每个簇内的未评价数据进行估值填充，以缓解评分数据稀疏性问题；在此基础上进一步利用用户兴趣偏好矩阵将具有相同兴趣偏好的用户聚类；最后，在每个用户类簇中应用基于用户的协同过滤推荐算法进行推荐。

本文在已有研究的基础上，针对上述问题展开研究，主要贡献包括：

(1)提出一种基于用户兴趣偏好的评分矩阵模型，用以查找与当前用户具有相似兴趣偏好的近邻用户，并将对项目类型具有相似兴趣偏好的用户进行聚类。

(2)提出一种基于项目间相似度的加权 Slope One 算法填充模型，对未评分项进行估值填充，并结合用户兴趣偏好矩阵实现基于用户的协同过滤推荐。

(3)在 MovieLens 数据集上进行算法对比实验，以验证本文所提出算法的有效性。

2 传统的协同过滤推荐算法及问题分析

2.1 相似性计算方法

协同过滤中常用的相似度计算方法有 3 种：余弦相似性、修正的余弦相似性以及皮尔逊相关相似性。

(1)余弦相似性。两个用户间的相似度可以看作是两个用户评分向量之间夹角的余弦值，余弦值越大，说明用户间的相似度越高。假设两个用户 x 和 y 的评分向量分别为 u 和 v，则 x 和 y 之间的余弦相似度 $sim(x,y)$ 为：

$$sim(x,y)=\cos(u,v)=\frac{u\cdot v}{\|u\|\cdot\|v\|} \tag{1}$$

(2)修正的余弦相似性。余弦相似性度量方法忽略了不同用户的评分尺度问题。修正的余弦相似性度量方法减去了用户对项目的平均评分。设 $I_{u,v}$ 表示用户 u 与用户 v 共同评分过的集合，I_u 和 I_v 分别表示用户 u 和 v 的评分集合。则用户 u 和 v 的修正余弦相似度为：

$$sim(u,v)=\frac{\sum\limits_{i\in I_{u,v}}(R_{u,i}-\bar{R}_u)(R_{v,i}-\bar{R}_v)}{\sqrt{\sum\limits_{i\in I_u}(R_{u,i}-\bar{R}_u)^2}\sqrt{\sum\limits_{i\in I_v}(R_{v,i}-\bar{R}_v)^2}} \tag{2}$$

(3)皮尔逊相关相似性。假设用户 u 和用户 v 的共同评分项集合为 $I_{u,v}$，$R_{u,i}$ 和 $R_{v,i}$ 分别表示用户 u 和用户 v 对项目 i 的评分。则用户 u 和 v 的皮尔逊相关相似性为：

$$sim(u,v)=\frac{\sum\limits_{i\in I_{u,v}}(R_{u,i}-\bar{R}_u)(R_{v,i}-\bar{R}_v)}{\sqrt{\sum\limits_{i\in I_{u,v}}(R_{u,i}-\bar{R}_u)^2}\sqrt{\sum\limits_{i\in I_{u,v}}(R_{v,i}-\bar{R}_v)^2}} \tag{3}$$

2.2 基于 K 近邻计算的评分预测及推荐

通常采用 K 近邻[9]方法进行评分预测，即选择与目标用户最相似的 K 个用户作为最近邻集合来进行计算。假设集合 U 表示目标用户 u 的最近邻集合，则用户 u 对项目 i 的预测评分值为：

$$p(u,i)=\bar{R}_u+\frac{\sum\limits_{u_k\in U}sim(u,u_k)\times(R_{u_k,i}-\bar{R}_{u,k})}{\sum\limits_{u_k\in U}sim(u,u_k)} \tag{4}$$

在完成评分预测后，取预测评分最高且不在目标用户已评价项目集合中的前 N 个项目作为 Top-N 推荐集，实施推荐。

2.3 协同过滤算法的问题分析

数据稀疏性问题是影响推荐准确性的核心问题之一。基于稀疏的评分矩阵进行相似度计算的误差非常大，推荐准确性也比较低。为了缓解这个问题，常用的方法是对稀疏矩阵中的未评分项进行填充，然而这些方法通常使用 0 或均值作为缺省值来进行填充，但实际应用中用户对不同项目的评分并非都是一致的，因此采用这种填充方式对算法推荐准确性的提高也是有限的。

基于以上分析，本文提出了一种基于项目间相似度的加权 Slope One 算法填充模型，在此基础上提出了一种融合聚类与用户兴趣偏好的协同过滤推荐算法。

3 融合聚类与用户兴趣偏好的协同过滤推荐算法

3.1 构建用户兴趣偏好矩阵

传统的协同过滤算法往往忽略对项目属性特征的考虑，我们认为通过加入对项目属性特征的考虑，能够获得更好的推荐效果。通常不同项目具有不同的类别属性，例如电影“Toy Story”就具有 educator，engineer 与 entertainment 3 个标签属性。

用户评价的项目所具有的标签属性通常能够在一定程度上反映用户对相应类型的偏好。比如有些用户偏爱具有喜剧特色的电影，而另一些用户偏爱恐怖元素的电影。考虑到这一特性，本文通过结合用户评价过的项目与不同项目所具有的项目标签属性，可以获得用户对不同项目类型的评价矩阵。考虑到用户对不同项目类型的偏好程度的差异性，本文利用式(5)对用户-项目评分矩阵进行重构，从而形成一个用户对项目类型的“用户兴趣偏好矩阵”(User-Type，UT)。

$$P(u)_x=\frac{\sum_{i\in I_{u,x}} R_{u,i}}{sum(I_{u,x})} \tag{5}$$

其中，$I_{u,x}$表示用户 u 所评价过的项目中具有 x 标签属性的项目集合，$R_{u,i}$表示用户 u 对项目 i 的真实评分，$sum(I_{u,x})$表示 $I_{u,x}$集合中的元素个数。

例如，表 1 所列的评分矩阵 S 中有用户 U_1，U_2，U_3 以及项目 I_1，I_2，I_3。表 2 所列的项目类型信息矩阵 P 中的项目拥有 4 种属性类别：T_a，T_b，T_c 和 T_d。

表 1　用户-项目评分矩阵 S

	I_1	I_2	I_3
U_1	3	4	4
U_2	?	2	?
U_3	4	?	3

表 2　项目类型信息矩阵 P

	T_a	T_b	T_c	T_d
I_1	1	1	1	0
I_2	0	1	1	1
I_3	1	1	0	0

在由矩阵 S 与 P 获取用户的偏好矩阵 UT 时，通常将 S 中的未评分数据项记为 0，即有：

$$R_{i,j}=\begin{cases} r_{i,j}, & \text{if user } i \text{ rated item } j \\ 0, & \text{if user } i \text{ not rated item } j \end{cases}$$

且在类别矩阵 P 中，当项目 I_j 具备属性 T_x 时，该项的评分记为 1，反之为 0，即有：

$$P_{j,x}=\begin{cases} 1, & \text{if item } j \text{ has attributes } x \\ 0, & \text{if item } j \text{ not has attributes } x \end{cases}$$

UT 矩阵可通过以下步骤构建：

Step1　获取用户评分数据。根据表 2，用户 U_1 评价过项目 $I_1=3$，$I_2=4$，$I_3=4$；用户 U_2 评价过项目 $I_2=3$；用户 U_3 评价过 $I_1=4$，$I_3=3$。

Step2　获取项目类型信息。参照表 3，I_1 具有 T_a，T_b，T_c 属性，I_2 具有 T_b，T_c，T_d 属性，I_3 具有 T_a，T_b 属性。

Step3　提取用户对项目类型的兴趣偏好。例如用户 U_1 在类型维度 T_b 上的取值可以由式(5)求得，即 $P_{1,b}=(3+4+4)/3=3.67$；同理，可以得到当前用户针对 T_a，T_c，T_d 的偏好值：$P_{1,a}=3.5$，$P_{1,c}=3.5$，$P_{1,d}=4$。

据此可以构造矩阵 S 与 P 的 UT，如表 3 所列。

表 3　用户兴趣偏好矩阵 $UT(m,x)$

	T_a	T_b	T_c	T_d
U_1	3.5	3.67	3.5	4
U_2	0	2	2	2
U_3	3.5	3.5	4	0

3.2　项目聚类

项目聚类的核心思想是将所有项目按照相似度划分为若干个类簇，同一个类簇中的项目具有较高的相似度。本文对项目集进行聚类的目的是在后续处理中应用改进的 Slope One 算法对用户-项目矩阵中的未评分项进行估值填充时，将整个项目空间压缩到若干个类簇中，避免簇间项目(与类簇中项目的相似度较低的项目)评分的影响，进一步提高填充的有效性。

K-Means 和 K-Medoids 是两种应用广泛的聚类算法[10]，其算法原理相似，主要区别在于聚类中心点的选取。K-Medoids 要求选取的聚类中心为数据集中的项目，而 K-Means 选取的聚类中心可以是计算出来的结果，并不一定是数据集中的项目。与 K-Means 相比，K-Medoids 的计算复杂度更高，计算量更大。因此，本文采用 K-Means 算法对项目进行聚类。

K-Means 算法聚类的结果是 K 个不同的项目聚类簇。K-Means 算法的过程如图 1 所示。

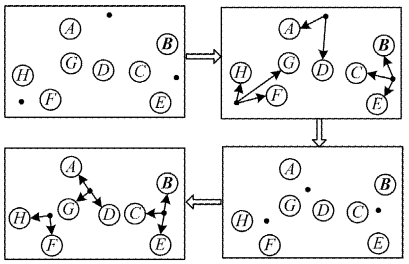

图 1　K-Means 算法的聚类过程

图 1 中，黑点代表聚类中心。K-Means 算法首先随机选取聚类中心，根据数据集的大小 N 选取合适的聚类个数 K；然后计算数据集中的对象和聚类中心的相似度，将每个对象分配到与其最相似的聚类中心所在的类中；接着计算每个聚类中所有对象的平均值，重新确定聚类中心；重复上述步骤，直到聚类中心不再变化。

K-Means 算法的具体步骤如下：

输入：用户表 Users、项目表 Items、评分表 Rating、聚类簇数 K
输出：K 个聚类
Step1　从用户表 Users 中查询所有 m 个用户，记为 U={u_1，u_2，…，u_m}
Step2　从项目列表 Items 中查询所有 n 个项目，记为 I={i_1，i_2，…，i_n}
Step3　初始化 k 个空聚类，记为 A={a_1，a_2，…，a_k}
Step4　随机选取 k 个项目评分向量作为 K 个初始聚类中心，记为 B={b_1，b_2，…，b_k}
Step5　Repeat
　For $i_i\in$I
　　For $b_j\in$B
　　计算项目 i_i 与 b_j 之间的相似度 sim(i_i，b_j)
　　sim (i_i，b_t)=max{sim(i_i，b_1)，sim(i_i，b_2)，sim(i_i，b_k)}
　　$a_t=a_t\cup i_i$
　For $a_i\in$A
　　For $u_j\in$U
　　计算用户 u_j 对聚类 a_i 中所有项目的平均评分并赋值给 b_i
　　Until 聚类中心 b_1，b_2，…，b_k 不再发生变化
Step6　返回

通过上述算法得到 K 个聚类后，只在同一个簇中对稀疏矩阵进行填充，不仅缩小了计算填充的范围，而且避免了簇外项目评分干扰造成的误差。

项目聚类个数 K 的取值是比较关键的。因为当 K 值偏小时，往往不能实现项目集的有效划分，簇内存在相似度较小、噪音较大的项目元素，进而给算法预测带来误差；而当 K 值偏大时，会导致有些簇内的元素个数较少，评分参照的项目数据不充足，同样会使算法的准确性受到影响。

3.3　基于项目相似度的加权 Slope One 算法填充模型

现有的填充方法(例如均值填充法、众数填充法等)同等对待用户的未评分项目，忽略了用户的偏好性，填充效果不理想。Daniel Lemire 教授在充分考虑了用户偏好性的前提下，

于 2005 年提出了 Slope One 算法，该算法通过对评分矩阵进行线性回归，预测矩阵中的空缺值。Slope One 算法通过线性回归的方法来将用户的偏好计入影响因子，从而计算填充评分矩阵，但是忽略了用户评分项的数量对计算目标评分项的贡献及项目之间的相似度对计算目标评分项的贡献。因此，本文在充分理解 Slope One 算法的基础上，基于相似度计算公式(2)，提出了基于项目相似度的加权 Slope One 算法。

基本 Slope One 算法的思想是：设用户 u 对某两个项目的评分为 x，y，该算法假设所有用户对这两个项目的评分 x 与 y 之间符合线性关系 $y=x+b$。基于已经对这两个项目评过分的大量用户评分数据来拟合该线性函数以获得参数 b 的估计值，最后将目标用户已有的项目评分 x 进行拟合，从而获得 u 对项目的评分估计值。

设 $U_{i,j}$ 为评价过项目 i 和 j 的用户集合，R_{uj} 为用户 u 对 j 的评分。如果一个项目被目标用户和其他用户同时评价过，那么将该项目加入到集合 I_i 中。$sum(H)$ 表示集合 H 中的元素个数，则传统的平均偏差 dev_{ij} 和预测评分 $P(u)_i$ 分别为：

$$dev_{ij}=\sum_{u\in U_{ij}}\frac{R_{ui}-R_{uj}}{sum(U_{ij})} \tag{6}$$

$$P(u)_i=\frac{\sum_{j\in I_i}dev_{ij}+R_{uj}}{sum(I_i)} \tag{7}$$

基本的 Slope One 算法忽视了不同用户数量的评分项所占的权重，在一定程度上会影响算法的准确性。例如，对 I_1 和 I_2 共同评分的用户数是 100，对 I_3 和 I_2 共同评分的用户数是 1000，预估用户 U_x 对项目 I_2 的评分值。考虑到权重，式(8)采用加权的 Slope One 算法(Weighted Slope One)来进行评分预测，即 $(100*(dev_{1,2}+R_{x,1})+1000(dev_{3,2}+R_{x,3}))/(100+1000)$。

$$Q(u)_i=\frac{\sum_{j\in I_i}(dev_{ij}+R_{uj})\cdot sum(U_{ji})}{\sum_{j\in I_i}sum(U_{ji})} \tag{8}$$

虽然 Weighted Slope One 算法通过不同用户评分数量加权对估值结果进行了修正，但是该算法并没有考虑项目间相似度差异对估值结果的影响。基于上述考虑，本文在 Weighted Slope One 算法的基础上引入了项目间相似度 $sim(i,j)$，并将其作为权值对算法的估值结果进行进一步修正，即基于项目相似度的加权 Slope one 算法的预测评分公式 $Q(u)_i$ 为：

$$Q(u)_i=\frac{\sum_{j\in I_i}(dev_{ij}+R_{uj})\cdot sum(U_{ji})\cdot sim(i,j)}{\sum_{j\in I_i}sum(U_{ji})\cdot sim(i,j)} \tag{9}$$

本文对 Slope One 算法进行改进的重要一点是，只在目标项目 I_x 所在的类簇 C_x 中进行评分计算而不是针对整个项目集，并且由于不同用户对项目的偏好也是不一致的，因此在获取用户评分时，采用全部的用户集往往也会给评估结果带来一定的误差。因此本文通过构建 UT 矩阵及计算用户间的相似度，来有效获取当前待评分项用户 u 的近邻用户 $H(u)$，则式(9)可改进为：

$$Q(u)_i=\frac{\sum_{j\in I_i,I_i\subseteq C(i),U\subseteq H(u)}(dev_{ij}+R_{uj})\cdot sum(U_{ji})\cdot sim(i,j)}{\sum_{j\in I_i,I_i\subseteq C(i),U\subseteq H(u)}sum(U_{ji})\cdot sim(i,j)} \tag{10}$$

近邻用户 $H(u)$ 的取值大小为 n，表示前 n 个用户中与当前用户 u 最相近的用户集合，显然 n 的取值也是至关重要的。

以表 4 为例，假设 I_1，I_2，I_3 属于同一个项目类簇，U_1，U_2 为目标用户 U_3 的近邻用户，需要计算 U_3 对 I_1 的评分并对矩阵进行填充。具体过程如下：

Step1 获取项目的偏差值。I_1 和 I_2 的平均偏差为((1−2)+(3−4))/2=−1，I_1 和 I_3 的平均偏差为(1−3)/1=−2。

Step2 获取项目间的相似度。根据式(2)可得项目 I_1 和 I_2 之间的修正余弦相似度为 $sim(I_1,I_2)=1$，I_1 与 I_3 之间的修正余弦相似度为 $sim(I_1,I_3)=0.5$。

Step3 根据评分用户数量与项目间的相似度加权计算估值。由 I_1 和 I_2 可得用户 U_3 对 I_1 的评分为 3+(−1)=2，由 I_1 和 I_3 可得用户 U_3 对项目 I_1 的评分为 5+(−2)=3。由于对 I_1 和 I_2 都评价过的用户有 2 个，对 I_1 和 I_3 都评价过的用户有 1 个，即数量权重分别为 2 和 1，且有相似度权重 $sim(I_1,I_2)=1$，$sim(I_1,I_3)=0.5$。利用本文改进的 Slope One 算法即式(10)可得 U_3 对 I_1 的评分为 $(2*2*1+3*1*0.5)/(2*1+1*0.5)=2.2$。

用此方法依次计算并完成所有空缺值的填充，最终生成一个新的矩阵 R'，结果如表 4 所列。

表 4 填充后的用户-项目评分矩阵 R'

	I_1	I_2	I_3
U_1	1	2	3
U_2	3	4	?(2)
U_3	?(2.2)	3	5

3.4 用户聚类及算法推荐

由 3.3 节获得了填充后的评分矩阵 R'。传统的基于用户的协同过滤算法在查找 k 近邻时往往都是在整个用户集空间中进行查找非常耗时且不准确，因为并不是用户集空间中的所有用户都与待推荐用户具有较高的兴趣偏好相似性。基于这一点，本文先应用 K-means 算法在用户兴趣偏好矩阵 UT 中对用户数据集进行聚类，将具有相似属性类型偏好的用户聚集在同一个类簇中，形成 k 个用户聚类簇 $U=\{U_1,U_2,\cdots,U_k\}$。然后在每一个用户类簇 $U_x\in U$ 中针对其待推荐用户查找 k 近邻集合，进而应用式(4)对原始评分矩阵进行评分预测并实施推荐。这样不仅大大提高了原始算法的时效性，同时也提高了评分预测的准确度。

3.5 算法执行过程

本文提出的融合聚类与用户兴趣偏好的协同过滤推荐算法的具体流程如图 2 所示。

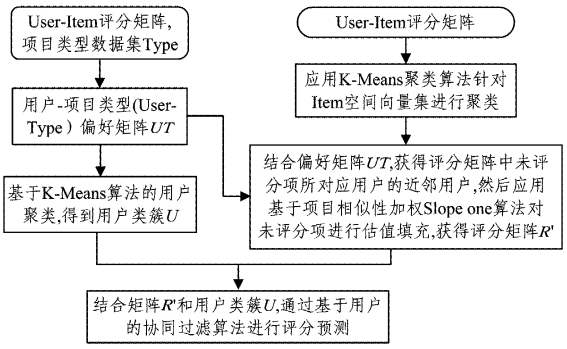

图 2 算法流程

Step1 首先通过用户-项目(User-item)评分矩阵 R 和项目类型数据集 $Type$，获取用户对不同项目类型(User-Type)的用户兴趣偏好矩阵 UT；

Step2 输入用户-项目(User-item)评分矩阵 R，通过 K-Means 聚类算法对项目数据集进行聚类，获得 k 个项目簇划

分 $I=\{I_1,I_2,\cdots,I_k\}$；

Step3　在每一个项目类簇 $I_i\in I$ 中，通过用户兴趣偏好矩阵 C 获取待估值项所对应用户的近邻用户，进而采用基于项目相似度的加权 Slope One 算法对 I_i 中的未评分项进行估值填充，获得新的 User-item 矩阵 R'；

Step4　在 UT 矩阵中，针对用户数据集，应用 K-Means 聚类算法将具有相似用户兴趣偏好的用户划分到同一个用户类簇中，得到用户簇划分 $U=\{U_1,U_2,\cdots,U_k\}$；

Step5　结合填充后的 User-item 矩阵 R' 与用户簇划分 U，在每一个类簇中应用相似度计算式(2)查找当前用户的若干个最近邻居，进而根据式(4)进行评分预测，并实施 Top-N 推荐。

4　实验比较和分析

4.1　实验环境与设置

本文的实验环境为一台 PC 机，机器的配置为 Intel Core 2 Duo、2.66GHz 4GB，操作系统为 Windows 7，实验中的所有程序使用 Python 实现。本文选择 Apache Mahout[11] 作为基础推荐引擎，该开源项目已经集成了协同过滤、聚类和分类等多种推荐算法。本文在 Mahout 的基础上开发部署提出的算法，并选择 MovieLens 数据集[12] 作为实验的测试数据集。MovieLens 是明尼苏达大学 GroupLens 小组搜集的电影评价数据，通过用户对电影的评分(1～5)进行电影推荐。MovieLens数据集提供了 4 种量级的数据：10 万条评分、100 万条评分、1000 万条评分、2000 万条评分。实验中选取第一个量级的数据集，该数据集中包含了 943 个用户针对 1682 部电影的 100000 个真实评分，评分区间为[1,5]，分数越高表示用户对该部电影的喜爱程度越深，并且数据集中的每个用户至少对其中的 20 部电影进行了打分。数据集随机地划分为训练集和测试集两部分，其中训练集占整个数据集的 80%，测试集占 20%。全部的电影被划分到 Action，Adventure，Children's，Crime等 19 个类别中，该数据集的稀疏度为 93.7%。

4.2　评估标准

预测精度度量了预测评分和真实评分之间的逼近度。两个广泛使用的精度指标是 RMSE 和 MAE。为了考查预测值与真实值之间的差距，本文采用平均绝对偏差(MAE)来衡量协同过滤算法的准确性。MAE 值越小，表示算法的准确率越高。假定通过推荐算法预测的用户对项目的评分集合为 $P=\{p_1,p_2,\cdots,p_x\}$，用户对相应项目的真实评分记为 $Q=\{q_1,q_2,\cdots,q_x\}$，那么可以将该推荐系统的评价 MAE 记为：

$$MAE=\frac{\sum_{i=1}^{x}|p_i-q_i|}{x} \tag{11}$$

4.3　实验结果及分析

为了验证本文提出算法的有效性，进行 4 部分实验。

实验 1　项目数据集聚类簇数 K 变化时的 MAE

本文算法的第一步需要对项目数据集进行聚类，从而得到若干个项目簇划分，使得在同一个类簇中的项目具有较高的相似性。本实验中 K 的范围为 15 到 50，每次间隔 5，依次观察 K 的变化对推荐精度的影响，最后选出最佳的 K 值，实验结果如图 3 所示。可以看出，K 取值 35 时，推荐算法的 MAE 最小。因此，本文在对项目进行聚类时取聚类簇数 K 值为 35。

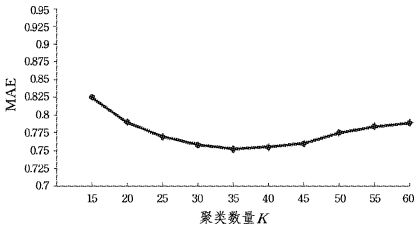

图 3　聚类簇数 K 变化时的 MAE

实验 2　估值填充情况下近邻用户数 n 变化时的 MAE

在项目类簇中，利用本文改进的 Slope One 算法对未评分项进行估值填充时，需要选取与当前待评分项所对应用户具有较高相似兴趣偏好的用户评分，即需要在用户兴趣偏好矩阵中求取当前用户的近邻用户，近邻用户数量对算法估值的准确度也起着关键的作用。本实验选取近邻数量 n 的范围为 15～50，每次间隔为 5，依次观察 n 的变化对本文算法推荐精度的影响，并选取推荐效果最好的 n 取值。实验效果如图 4 所示。

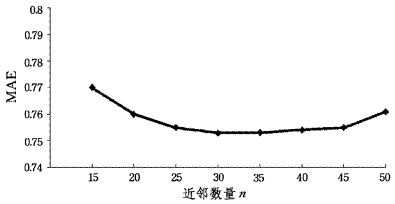

图 4　近邻数量 n 变化时的 MAE

由实验可知，当所取的近邻用户数在 30 附近时，本文算法预测评分的 MAE 达到最低，即算法的推荐效果最优，故本文选取近邻用户数 $n=30$。

实验 3　估值填充模型的有效性验证

本文基于项目相似度的加权 Slope One 算法填充模型基于原始 Weight Slope One 算法，在以下两方面进行了改进。

(1)评分数据的选取。本文将相似度较高的项目进行聚类，在选取用户评分时基于所构建的用户兴趣偏好矩阵模型，选取与当前待估值项所对应用户具有较高相似用户兴趣偏好的前 n 个近邻用户的评分数据。

(2)将项目间的相似度作为权值，对算法进行修正。

1)基于项目相似度加权的 Slope One 算法的有效性验证

实验分别用本文改进的 Slope One 算法和原始的 Weight Slope One 算法在全局评分数据空间对稀疏的用户-项目评分矩阵进行估值填充，并利用填充后的评分矩阵进行基于用户的协同过滤算法推荐，比较两种算法的 MAE，近邻数量 k 在 5～25 之间取值，依次间隔为 5，并观察 MAE 的变化。实验结果如图 5 所示，可以看出采用本文提出的基于项目相似度加权的 Slope One 算法对稀疏矩阵进行填充得到的 MAE 明显低于 Weighe Slope One 算法的 MAE，说明本文基于项目间相似度的加权 Slope One 算法是有效的。

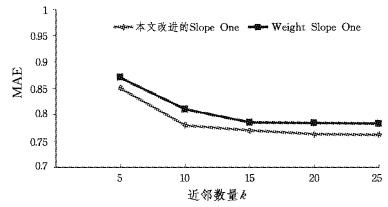

图 5　不同算法预测准确度的比较

2)项目聚类与估值填充时近邻用户的选取

实验基于改进的 Slope One 算法，分别采用在项目类簇中选取近邻用户的评分数据与原始的所有用户评分数据进行估值填充。利用填充后的矩阵，结合基于用户的协同过滤算法实施评分预测，比较两种方法的 MAE。实验选取项目聚类数量 $K=35$，估值填充时选取近邻用户数量 $n=30$ 进行实验，协同过滤推荐时近邻数量 k 取值范围为 5～25，依次间隔为 5，并观察 MAE 的变化。实验结果如图 6 所示，可以看出选取项目类簇中近邻用户的评分数据对未评分项的估值精确度比在整个评分数据空间进行估值的精确度更高。

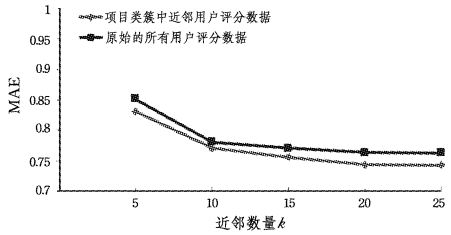

图 6 不同评分数据域选取的比较

实验说明，本文提出的填充模型能够显著降低 MAE，并提高推荐精度。

实验 4 与其他协同过滤算法的推荐精度对比

为了评价本文算法的推荐准确率，本实验将其与另外 3 种协同过滤推荐算法进行了对比，包括传统的基于用户的协同过滤算法(User-base CF)、基于项目的协同过滤算法(Item-base CF)以及文献[13]基于用户属性和评分的协同过滤推荐算法。本实验选取项目聚类个数 $K=35$，估值填充时取近邻用户大小 $n=30$，并利用用户兴趣偏好矩阵对用户进行聚类。考虑到算法时效性与准确性，实验选取用户聚类数量为 15，对各算法均采用修正的余弦相似性公式计算用户相似性，协同过滤推荐时近邻个数 k 取值范围为 5～35，每次间隔 5，依次观察 k 的变化对推荐精度的影响。实验结果如图 7 所示。

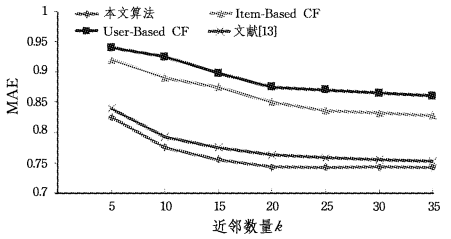

图 7 推荐精度的比较

可以看出，4 种算法的推荐精度均在一定程度上随着近邻个数 k 的增加而有所提升。相较于其他 3 种算法，本文所提出的算法能显著降低 MAE，且在近邻个数 k 接近 25 时本文算法的推荐效果达到最优，本文算法的 MAE 相对于 User-base CF、Item-base CF 和文献[13]分别提高了 14.38%，10.76%和 1.19%。由此可见，本文融合聚类与用户兴趣偏好的协同过滤算法具有更好的推荐准确性，能够有效缓解数据稀疏性问题。

结束语 本文提出了一种融合聚类和用户兴趣偏好的协同过滤推荐算法，用以解决数据稀疏性问题导致的推荐质量下降的问题。同时，提出了一种基于用户兴趣偏好的评分矩阵模型；采用基于项目的聚类算法对用户-项目评分矩阵进行聚类，并在聚类后的每一个簇中针对其中的待估值项，在用户兴趣偏好矩阵中查找当前用户的近邻用户，然后应用改进的 Slope One 算法对未评分数据进行估值填充。在此基础上，进一步通过用户兴趣偏好矩阵将具有相似兴趣偏好的用户进行聚类，然后结合填充矩阵在每一个用户类簇中应用基于用户的协同过滤算法实施推荐。实验结果表明，本文提出的算法能有效提高推荐质量。

参 考 文 献

[1] ADOMAVICIUS G, TUZHILIN A. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions [J]. IEEE Transactions on Knowledge and Data Engineering, 2005, 17(6): 734-749.

[2] 黄创光，印鉴，汪静，等. 不确定近邻的协同过滤推荐算法[J]. 计算机学报，2010，33(8)：1370-1377.

[3] 孟祥武，刘树栋，张玉洁，等. 社会化推荐系统研究[J]. 软件学报，2015，26(6)：1356-1372.

[4] 张锋，孙雪冬，常会友，等. 两方参与的隐私保护协同过滤推荐研究[J]. 电子学报，2009，37(1)：84-89.

[5] 吴湖，王永吉，王哲，等. 两阶段联合聚类协同过滤算法[J]. 软件学报，2010，21(5)：1042-1054.

[6] 韦素云，业宁，朱健，等. 基于项目聚类的全局最近邻的协同过滤算法[J]. 计算机科学，2012，39(12)：149-152.

[7] HUANG Y X, CUI B, ZHANG W Y, et al. TencentRec: Real-time stream recommendation in practice [C] // Proceedings of ACM SIGMOD International Conference on Management of Data. 2015: 227-238.

[8] 郭兰杰，梁吉业，赵兴旺. 融合社交网络信息的协同过滤推荐算法[J]. 模式识别与人工智能，2016，29(3)：281-288.

[9] 罗辛，欧阳元新，熊璋，等. 通过相似度支持度优化基于 K 近邻的协同过滤算法[J]. 计算机学报，2010，33(8)：1437-1445.

[10] HAN J W, KAMBER M. Data Mining: Concepts and Techniques [M]. San Francisco: Morgan Kaufmann Publishers, 2006.

[11] Apache Mahout [EB/OL]. [2016-09-23] http: mahout. apache. org.

[12] MovieLens datasets [EB/OL]. [2016-11-03] http://grouplens. org/datasets/movielens.

[13] 丁少衡，姬东鸿，王路路. 基于用户属性和评分的协同过滤推荐算法[J]. 计算机工程与设计，2015(2)：487-491.