计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200300072

融合用户评分与显隐兴趣相似度的协同过滤推荐算法

武建新　张志鸿
郑州大学信息工程学院　郑州 450001
(zzu_jxw@163.com)

摘　要　协同过滤算法是推荐系统中使用最广泛的算法,其核心是利用某兴趣爱好相似的群体来为用户推荐感兴趣的信息。传统的协同过滤算法利用用户-项目评分矩阵计算相似度,通过相似度寻找用户的相似群体来进行推荐,但是由于其评分矩阵的稀疏性问题,对相似度的计算不够准确,这间接导致推荐系统的质量下降。为了缓解数据稀疏性对相似度计算的影响并提高推荐质量,提出了一种融合用户评分与用户显隐兴趣的相似度计算方法。该方法首先利用用户-项目评分矩阵计算用户评分相似度;然后根据用户基本属性与用户-项目评分矩阵得出项目隐性属性;之后综合项目类别属性、项目隐性属性、用户-项目评分矩阵和用户评分时间,得到用户显隐兴趣相似度;最后融合用户评分相似度和用户显隐兴趣相似度得到用户相似度,并以此相似度寻找用户的相似群体以进行推荐。在数据集 Movielens 上的实验结果表明,相比传统算法中仅使用单一的评分矩阵来计算相似度,提出的新相似度计算方法不仅能够更加准确地寻找到用户的相似群体,而且还能够提供更好的推荐质量。

关键词:协同过滤;用户评分;显隐兴趣;项目隐性属性;用户基本属性
中图法分类号　TP301

Collaborative Filtering Recommendation Algorithm Based on User Rating and Similarity of Explicit and Implicit Interest

WU Jian-xin and ZHANG Zhi-hong
School of Information and Engineering,Zhengzhou University,Zhengzhou 450001,China

Abstract　Collaborative filtering algorithm is the most widely used algorithm in recommendation system. Its core is to use a group with similar interests to recommend information of interest for users. The traditional collaborative filtering algorithm uses the user item scoring matrix to calculate the similarity,and finds the similar groups of users through the similarity to recommend. However,due to the sparsity of the scoring matrix,the calculation of the similarity is not accurate enough,which indirectly leads to the degradation of the quality of the recommendation system. In order to alleviate the impact of data sparsity on the similarity calculation and improve the quality of recommendation,a similarity calculation method is proposed,which integrates user rating and user's explicit and implicit interest. This method first uses the user item scoring matrix to calculate the similarity of user's scoring,then uses the basic attribute of user and the user item scoring matrix to get the implicit attribute of project,then integrates the attribute of project category,the implicit attribute of project,the scoring matrix of user item and the scoring time of user to get the similarity of user's explicit and implicit interest,finally integrates the similarity of user's scoring and the similarity of user's explicit and implicit interest to find similar groups of users for recommendation. Experimental results on the data set Movielens show that compared with the traditional algorithm,which only uses a single scoring matrix to calculate the similarity,the new similarity calculation method can not only find similar groups of users more accurately,but also provide a better recommendation quality.

Keywords　Collaborative filtering,User rating,Explicit and implicit interest,Item implicit attribute,User basic attributes

1　引言

随着第四次工业革命的发展,人们可获取到的数据量呈爆炸性增长,导致了"信息过载"的问题,这使得大量无关的、冗余的信息阻碍了人们的选择,因此越来越多的学者开始研究推荐算法来解决这个问题,主要研究的推荐算法有基于规则的推荐、基于协同过滤的推荐、基于内容的推荐、基于社交网络的推荐、混合推荐[1]。协同过滤推荐算法是这些算法中使用最广泛和最成功的算法,它分为基于内存的协同过滤推荐和基于模型的协同过滤推荐。基于内存的协同过滤推荐又

到稿日期:2020-03-13　返修日期:2020-06-27　本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(11501523)
This work was supported by the National Natural Science Foundation of China (11501523).
通信作者:张志鸿(iezhzhang@zzu.edu.cn)

分为基于用户的推荐和基于项目的推荐，其通过用户的历史行为数据寻找兴趣爱好相似的群体，再根据相似群体的历史行为对用户进行推荐；基于模型的推荐则是根据用户的历史行为数据，将用户和项目映射为同一维度空间中的向量，通过计算内积得到用户对项目的喜好程度，然后进行推荐[2]。由于主要根据用户的历史行为数据进行推荐，协同过滤推荐算法面临的主要问题为数据的稀疏性和冷启动[3]。

基于模型的协同过滤推荐算法的核心是构建出用户和项目的抽象特征向量，主要是通过基于机器学习的思想进行构建，常用的有基于神经网络的构建方法[4]、基于奇异值分解的构建方法[5]、基于矩阵分解的构建方法[6]等。

基于内存的协同过滤推荐算法能否提高推荐质量，关键在于能否寻找到准确的相似群体，而寻找相似群体的关键则是能否准确计算用户或项目之间的相似度，因此，改进相似度的计算方式以提高推荐质量是近年来基于内存的协同过滤推荐算法的研究热点之一。传统计算相似度的方法有余弦相似度(Cosine Similarity，COS)、皮尔逊相似度(Pearson Correlation Coefficient，PCC)、修正的余弦相似度(Adjusted Cosine Similarity，ACOS)等，这些方法仅利用用户-项目评分矩阵计算相似度，由于数据的稀疏性问题，其推荐质量较低。为了改进相似度的计算方法，文献[7]提出了一个新的启发式相似度计算模型(New Heuristic Similarity Model，NHSM)，该模型将用户相似度分为3个因素：邻近性、重要性、奇异性，分别求取这3个因素的相似度并进行融合，结果表明该方法相比传统相似度方法有更好的表现。文献[8]提出了一个高效的相似度计算方法，命名为共鸣相似度(Resonance Similarity，RES)，该方法将用户的相似度分为一致性因素、距离因素和杰卡德因素，然后将这些因素以数学乘积的形式进行融合，并在6个真实的数据集上进行实验。结果表明，相比传统相似度计算方法，该方法的相似度计算表现有明显提升。文献[9]提出了一种新的相似度计算方法，其充分利用评分上下文信息，计算出项目的奇异因子，然后结合奇异因子和用户评分习惯求得相似度，其结果有效地提高了推荐系统的准确性。这些方法均是以用户-项目评分数据为基础，延伸出多个相似度因素以进行融合，虽然其结果有所改善，但是计算的相似度仍然不够准确。

除了用户-项目评分矩阵的数据可以反映用户的相似度外，根据用户评分项目的类别属性也可以挖掘出用户的偏好。文献[10]分别考虑了用户在评分习惯和项目选择上的差异性，融合了用户评分相似度和项目类别属性相似度，其结果相比传统算法也有明显的提高。文献[11]提出了一种基于项目评分和属性的相似度计算方法，该方法对评分相似度和属性相似度进行加权融合，其结果不仅在一定程度上解决了项目冷启动的问题，而且还提高了推荐的质量。这些方法通过引入项目的类别属性来挖掘用户对项目类别的兴趣，并结合用户-项目评分数据计算出相似度，虽然其相似度的准确性有所提高，但是忽略了时间因素对用户兴趣的影响。

用户的兴趣会随着时间的变化而变化，用户对项目的评分时间距当前时间越近，其评分数据越具有价值，越能反映出用户的兴趣，因此充分考虑时间的影响也可以提高相似度的准确性。文献[12]综合考虑了时间特征、评分差异、项目属性对相似度计算的影响，提出了一种基于评分差异和用户兴趣相似度的协同过滤算法，其结果比传统方法具有更好的精确度，能够更加准确地找到相似用户。文献[13]引入置信机制和时间权重以优化协同过滤算法，首先将原始评分数据加上时间权重，然后结合加权后的数据和置信机制计算出相似度，提高了推荐的准确性。

以上研究方法虽然考虑了用户-项目评分矩阵、项目类别属性、用户评分时间对相似度的影响，但是都没有考虑到用户的基本属性对相似度的影响，而这些属性也可以反映出用户的相似度。因此本文在以上数据的基础上引入用户基本属性数据，提出了一种融合用户评分与显隐兴趣相似度的协同过滤推荐算法。首先，由用户-项目评分矩阵得到用户评分相似度；然后，将用户属性进行离散化处理，结合用户-项目评分数据归纳出项目隐性属性；再由项目隐性属性、项目类别属性、用户评分时间、用户-项目评分矩阵得到用户显隐兴趣相似度；最后，综合考虑用户评分相似度和用户显隐兴趣相似度，找到目标用户的相似群体，并利用该群体的用户-项目评分数据对目标用户进行推荐。

2 传统的相似度计算方法

传统的协同过滤算法中常用的相似度计算方法有3种：余弦相似度、皮尔逊相似度、修正的余弦相似度。它们都是基于用户-项目评分矩阵来获取用户的相似度。

2.1 用户-项目评分矩阵

将用户-项目评分看作是一个矩阵 $\boldsymbol{R}_{m\times n}$(见表1)。矩阵共 m 行 n 列，m 表示用户的数量，用 $U=\{u_1,u_2,u_3,\cdots,u_m\}$ 表示用户集合；n 表示项目的数量，用 $V=\{v_1,v_2,v_3,\cdots,v_n\}$ 表示项目集合。矩阵中 r_{ij} 表示第 i 个用户 u_i 对第 j 个项目 v_j 的评分，假设评分值的范围是[1,5]，如果用户没有对项目评分，则默认该值为0。

表1 用户-项目评分矩阵

Table 1 User-item rating matrix

	v_1	v_2	…	v_j	…	v_n
u_1	r_{11}	r_{12}	…	r_{1j}	…	r_{1n}
u_2	r_{21}	r_{22}	…	r_{2j}	…	r_{2n}
…	…	…	…	…	…	…
u_i	r_{i1}	r_{i2}	…	r_{ij}	…	r_{in}
…	…	…	…	…	…	…
u_m	r_{m1}	r_{m2}	…	r_{mj}	…	r_{mn}

2.2 余弦相似度

将用户-项目评分矩阵的每一行看作用户在 n 维空间上的评分向量，则余弦相似度就是计算两个用户评分向量之间夹角的余弦值。其计算公式如式(1)所示：

$$\mathrm{sim}(u_i,u_j)^{\mathrm{COS}}=\frac{\overrightarrow{\boldsymbol{r}_i.}\cdot\overrightarrow{\boldsymbol{r}_j.}}{\|\overrightarrow{\boldsymbol{r}_i.}\|\cdot\|\overrightarrow{\boldsymbol{r}_j.}\|} \tag{1}$$

其中，$\overrightarrow{\boldsymbol{r}_i.}$ 是用户 u_i 的评分向量(即用户-项目评分矩阵中的第 i 行)，$\overrightarrow{\boldsymbol{r}_j.}$ 是用户 u_j 的评分向量(即用户项目评分矩阵中的第 j 行)。

2.3 皮尔逊相似度

皮尔逊相似度要求两个用户之间必须有共同评分的项

目。设用户 u_i 评分的项目集合为 I_i，用户 u_j 评分的项目集合为 I_j，k 为两个项目集合的交集（即 $k\in I_i\cap I_j$），则它们之间的皮尔逊相似度的计算公式如式(2)所示：

$$\mathrm{sim}(u_i,u_j)^{\mathrm{PCC}}=\frac{\sum_{e\in k}(r_{ie}-\overline{r_i})(r_{je}-\overline{r_j})}{\sqrt{\sum_{e\in k}(r_{ie}-\overline{r_i})^2}\sqrt{\sum_{e\in k}(r_{je}-\overline{r_j})^2}} \tag{2}$$

其中，$\overline{r_i}$ 为用户 u_i 的评分均值，$\overline{r_j}$ 为用户 u_j 的评分均值。

2.4 修正的余弦相似度

余弦相似度没有考虑用户评分的个性化差异，而修正的余弦相似度在原有公式的基础上进行调整，将用户评分减去用户对所有评分项目的平均值，以此来解决不同用户评分尺度的问题。设用户 u_i 评分的项目集合为 I_i，用户 u_j 评分的项目集合为 I_j，k 为两个项目集合的交集（即 $k\in I_i\cap I_j$），则它们的修正余弦相似度的计算公式如式(3)所示：

$$\mathrm{sim}(u_i,u_j)^{\mathrm{ACOS}}=\frac{\sum_{e\in k}(r_{ie}-\overline{r_i})(r_{je}-\overline{r_j})}{\sqrt{\sum_{e\in I_i}(r_{ie}-\overline{r_i})^2}\sqrt{\sum_{e\in I_j}(r_{je}-\overline{r_j})^2}} \tag{3}$$

其中，$\overline{r_i}$ 为用户 u_i 的评分均值，$\overline{r_j}$ 为用户 u_j 的评分均值。

2.5 传统相似度的缺点

上文简单介绍了 3 种传统的相似度计算方法，它们的准确性要求两个用户之间含有足够多的共同评分项目。但是由于数据的稀疏性，两个用户之间的共同评分项目很少，甚至没有，这就导致上述算法的准确性降低。例如：假设用户 u_i 对项目的评分向量为(1,1,2,4,0)，用户 u_j 对项目的评分向量为(0,0,2,0,3)，其中 0 表示没有对该项目进行评分。直观上看，这两个用户的相似度应该很小。如果对 PCC 相似度进行计算，则会出现分母为零的情况，其结果就不具有意义；而如果对 COS 和 ACOS 相似度进行计算，其结果值会很高，这就与真实情况相反。由此可见，在稀疏数据中，使用传统相似度计算方法会使相似度的准确性降低。为了缓解数据稀疏性对相似度计算的影响，本文提出了一种新的相似度计算方法。

3 融合用户评分与显隐兴趣的相似度计算方法

3.1 用户评分相似度

根据用户-项目评分矩阵的数据特点，本文将用户评分相似度划分为 3 个部分，分别为奇异相似度、调整的杰卡德相似度、评分差异相似度，将三者通过数学乘积的形式进行融合以便得到用户评分相似度。下面分别介绍这 3 个相似度的计算方法。

3.1.1 SIGMOID 函数

由于在计算用户评分相似度时需要将 3 个相似度的计算结果进行融合，需要将 3 个相似度的计算结果映射到同一个区间，而相似度的范围一般在 0 到 1 之间，为了将原始的相似度计算结果映射到这个区间，本文引入 SIGMOID 函数作为映射函数，其计算公式如式(4)所示：

$$f_{\mathrm{sig}}(x)=\frac{1}{1+e^{-x}} \tag{4}$$

其中，x 为原始的相似度结果。

3.1.2 奇异相似度

假设用户对项目的评价被分为两种：喜欢和不喜欢，则对于某一个项目，对其进行评价的用户集合可以被划分为喜欢该项目的集合和不喜欢该项目的集合，如果这两个集合的用户数量相差很大，例如有 90% 的用户喜欢该项目，10% 的用户不喜欢该项目，那么这 10% 的用户之间的相似度应该大于这 90% 的用户之间的相似度，这就是用户评价的奇异性，其中 10% 是该项目的正奇异因子，90% 是该项目的负奇异因子。为了满足该假设所提条件，将对项目评分为 4 和 5 的用户划分到喜欢该项目的集合中，将对项目评分为 1，2 和 3 的用户划分到不喜欢该项目的集合中。基于以上说明，对一些符号进行定义，如表 2 所列。

表 2 符号定义表

Table 2 Symbol description

符号	定义	描述
M	$\{1,2,3,\cdots,m\}$	用户编号集合
I	$\{1,2,3,\cdots,n\}$	项目编号集合
R^P	$\{4,5\}$	喜欢某项目的评分集合
R^N	$\{1,2,3\}$	不喜欢某项目的评分集合
P_i	$\{m\mid r_{mi}\in R^P\}$	喜欢项目 v_i 的用户编号集合
N_i	$\{m\mid r_{mi}\in R^N\}$	不喜欢项目 v_i 的用户编号集合
S_P^i	$1-\lvert P_i\rvert/\lvert M\rvert$	项目 v_i 的正奇异因子
S_N^i	$1-\lvert N_i\rvert/\lvert M\rvert$	项目 v_i 的负奇异因子

下面通过一个具体例子来模拟各个符号的取值。设用户-项目评分矩阵如表 3 所列。

表 3 用户-项目评分矩阵用例

Table 3 User-item rating matrix example

	v_1	v_2	v_3	v_4	v_5
u_1	3	**5**	**5**	2	1
u_2	**4**	3	**4**	1	**5**
u_3	5	2	2	4	1
u_4	2	**4**	**4**	2	3
u_5	1	**4**	**5**	2	**5**

由表 2 可知，用例中各个符号的值如表 4 所列。

表 4 用例符号取值表

Table 4 Symbolic values of the example

符号	取值
M	$\{1,2,3,4,5\}$
I	$\{1,2,3,4,5\}$
R^P	$\{4,5\}$
R^N	$\{1,2,3\}$
P_i	$P_1=\{2,3\},P_2=\{1,4,5\},P_3=\{1,2,4,5\},$ $P_4=\{3\},P_5=\{2,5\}$
N_i	$N_1=\{1,4,5\},N_2=\{2,3\},N_3=\{3\},$ $N_4=\{1,2,4,5\},N_5=\{1,3,4\}$
S_P^i	$S_P^1=1-2/5=3/5,S_P^2=1-3/5=2/5,S_P^3=1-4/5=1/5,$ $S_P^4=1-1/5=4/5,S_P^5=1-2/5=3/5$
S_N^i	$S_N^1=1-3/5=2/5,S_N^2=1-2/5=3/5,S_N^3=1-1/5=4/5,$ $S_N^4=1-4/5=1/5,S_N^5=1-3/5=2/5$

由表 4 可知每个项目的正负奇异因子，为了获得用户的奇异相似度，需要根据用户对项目的评分组合，将两个用户之间共同评分的项目划分为 3 组，再结合项目的正负奇异因子计算出各个组中项目的奇异因子，具体划分及奇异因子计算如表 5 所列。

表 5 项目划分定义

Table 5 Item division definition

项目组别	评分组合	奇异因子
A	(喜欢，喜欢)	$S_P^i\cdot S_P^i$
B	(不喜欢，不喜欢)	$S_N^i\cdot S_N^i$
C	(喜欢，不喜欢)、(不喜欢，喜欢)	$S_P^i\cdot S_N^i$

由表5可知，A组的项目为两个用户同时喜欢的项目，B组的项目为两个用户同时不喜欢的项目，C组的项目则为两个用户既不同时喜欢也不同时不喜欢的项目。至此，便可根据划分的项目组别和奇异因子计算出用户的奇异相似度。设用户 u_i 和用户 u_j 的共同评分项目集合为 U，则可得到用户 u_i 和用户 u_j 的奇异相似度计算公式如式(5)所示：

$$sim(u_i,u_j)^S=\frac{|A|L(A)+|B|L(B)+|C|L(C)}{|U|} \tag{5}$$

$$L(A)=\frac{\sum_{e\in A}[(1-2(f_{\text{sig}}(|r_{ie}-r_{je}|)-0.5))S_P^eS_P^e]}{\sum_{e\in A}S_P^eS_P^e} \tag{6}$$

$$L(B)=\frac{\sum_{e\in B}[(1-2(f_{\text{sig}}(|r_{ie}-r_{je}|)-0.5))S_N^eS_N^e]}{\sum_{e\in B}S_N^eS_N^e} \tag{7}$$

$$L(C)=\frac{\sum_{e\in C}[(1-2(f_{\text{sig}}(|r_{ie}-r_{je}|)-0.5))S_P^eS_N^e]}{\sum_{e\in C}S_P^eS_N^e} \tag{8}$$

式(5)按照项目划分定义将用户共同评分项目集合划分到A,B,C组。首先利用每个组中项目的奇异因子和用户评分得到各个组的用户相似度，然后根据各组所占比例进行加权，最后求得用户的奇异相似度。

3.1.3 调整的杰卡德相似度

由于数据的稀疏性，用户共同评分的项目可能非常少，甚至没有，需要考虑用户之间共同评分的项目数量。设用户 u_i 评分的项目数量为10，用户 u_i 和用户 u_j 共同评分的项目数量为5，占用户 u_i 评分项目数量的50%；用户 u_i 和用户 u_k 共同评分的项目数量为8，占用户 u_i 评分项目数量的80%，认为用户 u_i 和用户 u_k 的相似度应该大于用户 u_i 和用户 u_j 的相似度，因此调整的杰卡德相似度的公式如式(9)所示：

$$\text{sim}(u_i,u_j)^C=1-e^{-\omega p} \tag{9}$$

$$p=\frac{|I_i\cap I_j|}{|I_i|} \tag{10}$$

式(10)中的 I_i 是用户 u_i 的评分项目集合，I_j 是用户 u_j 的评分项目集合。式(9)中的 ω 是调节参数，本文假设当用户 u_i 与某个用户共同评分的项目数量等于用户 u_i 与其他所有用户的共同评分项目数量的均值时，对应的调整杰卡德相似度为0.5，由此计算出 ω 的值。

3.1.4 评分差异相似度

每个用户之间存在着个性化差异，有些用户爱好评高分，有些用户爱好评低分，因此为了使相似度更加准确，首先使用用户评分减去用户对所有项目的评分均值，并用其表示用户的真实评分，然后根据真实评分的差值计算出评分差异相似度。其计算公式如式(11)所示：

$$\text{sim}(u_i,u_j)^D=2\left(1-f_{\text{sig}}\left(\frac{\sum_{e\in U}|(r_{ie}-\bar{r}_i)-(r_{je}-\bar{r}_j)|}{|U|}\right)\right) \tag{11}$$

其中，U 为用户 u_i 和 u_j 的共同评分项目集合，$|U|$ 是用户 u_i 和 u_j 的共同评分项目个数。结合式(5)、式(9)、式(11)可以得到用户的评分相似度，如式(12)所示：

$$\text{sim}(u_i,u_j)^R=\text{sim}(u_i,u_j)^S\cdot\text{sim}(u_i,u_j)^C\cdot\text{sim}(u_i,u_j)^D \tag{12}$$

3.2 用户显隐兴趣相似度

第3.1节主要使用用户-项目评分矩阵构建用户评分相似度，本节将首先结合用户基本属性和用户-项目评分矩阵构建项目隐性属性，然后基于项目隐性属性、项目类别属性和评分时间计算出用户显性兴趣相似度和用户隐性兴趣相似度，最后融合二者得到用户显隐兴趣相似度。

3.2.1 时间权重

为了能够准确表示用户最新的兴趣，本文加入时间间隔函数以计算用户-项目评分的时间权重，用户评分项目的时间距当前时间越近，该用户-项目评分的时间权重就越大。设所有用户对所有项目的评分时间与当前时间的最大间隔为 T，用户 u_i 对项目 v_j 的评分时间与当前时间的间隔为 t，则用户 u_i 对项目 v_j 评分的时间权重如式(13)所示：

$$f_{time}(t)=e^{-(\frac{t}{T})} \tag{13}$$

3.2.2 项目隐性属性基础构建方法

用户基本属性主要包括姓名、性别、年龄、职业、学历等相对容易获取的数据。对于某一个项目，可以通过其用户-项目评分数据，获得评分该项目的用户集合的基本属性的取值分布，该分布可以进而转化为项目的隐性属性。下面以“性别”为例，说明如何将用户基本属性转化为项目隐性属性。设对项目 v_i 评分的用户集合为 U_i，用户基本属性“性别”的取值集合为{男性，女性}，将集合 U_i 按照用户“性别”属性划分为集合 M(男性集合)和集合 F(女性集合)，则可以得到对项目 v_i 评分的男性用户占对项目 v_i 评分的所有用户的比例，将该比例记为项目 v_i 的隐性属性，即男性化指数的取值，同理可得女性化指数的取值。若将该操作扩展到所有项目，即可得到所有项目关于用户基本属性“性别”的隐性属性取值。综上，定义项目的隐性属性“男性化指数”如式(14)所示：

$$I_M^i=\frac{|M|}{|U_i|} \tag{14}$$

其中，I_M^i表示项目 v_i 的隐性属性“男性化指数”，$|M|$为集合 M 中男性用户的个数，$|U_i|$为集合 U_i 中用户的个数。项目 v_i 的隐性属性“女性化指数”如式(15)所示：

$$I_F^i=\frac{|F|}{|U|} \tag{15}$$

同理可将用户的其他基本属性转化为项目的隐性属性。

3.2.3 改进的项目隐性属性构建方法

由第3.2.2节可知，当用户基本属性的取值为连续值时，根据基础构建方法，每个取值会对应一个项目隐性属性。因为连续属性的取值很多，所以会导致项目隐性属性的维度很大，进而造成维数爆炸，因此，在构建项目隐性属性之前，应先将用户的基本属性取值进行离散化。此外，每个项目的评分用户个数存在差异，若对某个项目评分的用户很少，则基本属性的取值分布就不能准确地表示该项目的隐性属性，因此根据项目的评分用户个数，在原始公式的基础上添加调节因子进行改进，具体改进如下。

设用户基本属性 A 的取值集合为$\{a_1,a_2,\cdots,a_n\}$，对项目 v_i 评分的用户集合为 U，则属性 A 对项目 v_i 产生的隐性属性计算公式如式(16)所示：

$$I_{a_j}^i=2\left(f_{\text{sig}}\left(\frac{|U|}{\lambda_1}\right)-\frac{1}{2}\right)\frac{|A_j|}{|U|} \tag{16}$$

其中，$j\in\{1,2,3,\cdots,n\}$，$I_{a_j}^i$表示项目的 a_j 化隐性属性，$|A_j|$

表示集合U中A属性值为a_j的用户个数，$|U|$表示集合U的用户个数，λ_1为调节参数，$2\left(f_{\text{sig}}\left(\frac{|U|}{\lambda_1}\right)-\frac{1}{2}\right)$为调节因子。本文假设所有项目的评分用户个数的中位数所对应的调节因子值为0.5(即当集合U的用户个数为所有项目的评分用户个数的中位数时对应的调节因子值为0.5)，由此得出λ_1的取值。

3.2.4 用户隐性兴趣相似度

构建项目隐性属性之后，可以得到项目隐性属性矩阵，如表6所列。其中，I_j表示项目第j个隐性属性；v_i表示第i个项目；I_j^i表示第i个项目的第j个隐性属性的取值，可通过式(16)计算得出。

表6 项目隐性属性矩阵

Table 6 Implicit genres matrix

	I_1	I_2	…	I_j	…	I_t
v_1	I_1^1	I_2^1	…	I_j^1	…	I_t^1
v_2	I_1^2	I_2^2	…	I_j^2	…	I_t^2
…	…	…	…	…	…	…
v_i	I_1^i	I_2^i	…	I_j^i	…	I_t^i
…	…	…	…	…	…	…
v_n	I_1^n	I_2^n	…	I_j^n	…	I_5^n

设$\vec{\boldsymbol{I}}^k.$为项目v_k的隐性属性向量，即项目隐性属性矩阵中的第k行。用户u_i评分的项目集合为N_i，用户u_i的隐性兴趣向量用$\vec{\boldsymbol{U}}_i$表示，其计算公式如式(17)所示：

$$\vec{\boldsymbol{U}}_i=2\left(f_{\text{sig}}\left(\frac{|N_i|}{\lambda_2}\right)-\frac{1}{2}\right)\frac{\sum\limits_{e\in N_i}(f_{time}(t_{ie})\cdot r_{ie}\cdot \vec{\boldsymbol{I}}^e.)}{\sum\limits_{e\in N_i}f_{time}(t_{ie})} \tag{17}$$

其中，t_{ie}表示用户u_i对项目v_e评分的时间与当前时间的间隔；r_{ie}表示用户u_i对项目v_e的评分，λ_2为调节参数；$2\left(f_{\text{sig}}\left(\frac{|N_i|}{\lambda_2}\right)-\frac{1}{2}\right)$为调节因子，表示若用户评分的项目个数少，则不足以表现出用户的兴趣。本文假设所有用户评分的项目个数的中位数所对应的调节因子值为0.5(即当用户评分的项目个数为所有用户评分的项目个数的中位数时，对应的调节因子值为0.5)，由此得出λ_2的取值。然后通过余弦相似度求出用户的隐性兴趣相似度，如式(18)所示：

$$\text{sim}(u_i,u_j)^{IMP}=\frac{\vec{\boldsymbol{U}}_i\cdot\vec{\boldsymbol{U}}_j}{\|\vec{\boldsymbol{U}}_i\|\|\vec{\boldsymbol{U}}_j\|} \tag{18}$$

3.2.5 用户显性兴趣相似度

第3.2.4节介绍了由项目隐性属性构建用户隐性兴趣相似度，本节将使用项目类别属性构建用户显性兴趣相似度。项目类别属性是开发者为其定义的，用以挖掘用户的兴趣。项目类别属性矩阵如表7所列。

表7 项目类别属性矩阵

Table 7 Explicit genres matrix

	E_1	E_2	…	E_j	…	E_t
v_1	E_1^1	E_2^1	…	E_j^1	…	E_t^1
v_2	E_1^2	E_2^2	…	E_j^2	…	E_t^2
…	…	…	…	…	…	…
v_i	E_1^i	E_2^i	…	E_j^i	…	E_t^i
…	…	…	…	…	…	…
v_n	E_1^n	E_2^n	…	E_j^n	…	E_t^n

表7中，E_j表示项目的第j个类别属性，E_j^i表示第i个项目的第j个类别属性值，该值为0或1，即表示不属于该类别或属于该类别。

设$\vec{\boldsymbol{E}}^k.$为项目v_k的类别属性向量，用户u_i评分的项目集合为N_i，将用户u_i的显性兴趣向量用$\vec{\boldsymbol{F}}_i$表示，则计算公式如式(19)所示：

$$\vec{\boldsymbol{F}}_i=2\left(f_{\text{sig}}\left(\frac{|N_i|}{\lambda_2}\right)-\frac{1}{2}\right)\frac{\sum\limits_{e\in N_i}(f_{time}(t_{ie})\cdot r_{ie}\cdot \vec{\boldsymbol{E}}^e.)}{\sum\limits_{e\in N_i}f_{time}(t_{ie})} \tag{19}$$

其中，λ_2，t_{ie}，r_{ie}同式(17)中的λ_2，t_{ie}，r_{ie}。

同式(18)可得用户的显性兴趣相似度如式(20)所示：

$$\text{sim}(u_i,u_j)^{EXP}=\frac{\vec{\boldsymbol{F}}_i\cdot\vec{\boldsymbol{F}}_j}{\|\vec{\boldsymbol{F}}_i\|\|\vec{\boldsymbol{F}}_j\|} \tag{20}$$

最后融合用户的显性兴趣相似度和用户的隐性兴趣相似度，可得到用户的显隐兴趣相似度如式(21)所示：

$$\text{sim}(u_i,u_j)^{I}=\beta\cdot\text{sim}(u_i,u_j)^{EXP}+(1-\beta)\cdot\text{sim}(u_i,u_j)^{IMP} \tag{21}$$

其中，权重β在实验中进行设定，其取值范围为[0,1]。

3.3 用户相似度

用户的相似度受到用户评分相似度和用户显隐兴趣相似度的影响，将第3.1节和第3.2节得到的两个相似度进行加权组合，便可得到用户的相似度，如式(22)所示：

$$\text{sim}(u_i,u_j)^{F}=\alpha\cdot\text{sim}(u_i,u_j)^{R}+(1-\alpha)\cdot\text{sim}(u_i,u_j)^{I} \tag{22}$$

其中，权重α在实验中进行设定，其取值范围同样是[0,1]。$\text{sim}(u_i,u_j)^F$越大，用户u_i和u_j的相似度越大。

3.4 用户评分预测

得出用户相似度后，首先寻找用户的k个最相似用户作为相似群体，然后根据式(23)对目标项目进行评分预测，最后将评分最高的项目推荐给用户。

$$r'_{ui}=\bar{r}_u+\frac{\sum\limits_{v\in U_k}(r_{vi}-\bar{r}_v)\cdot\text{sim}(u,v)^F}{\sum\limits_{v\in U_k}\text{sim}(u,v)^F} \tag{23}$$

其中，r'_{ui}表示用户u对项目i的评分预测，$\bar{r}_u$表示用户u的评分均值，U_k表示用户u的k个相似用户。

4 融合用户评分与显隐兴趣相似度的协同过滤算法

由第3节可知，融合用户评分与显隐兴趣相似度的协同过滤算法可以分为5个步骤：1)根据用户-项目评分矩阵计算用户评分相似度；2)根据用户基本属性和用户-项目评分矩阵构建项目隐性属性；3)根据项目隐性属性、项目类别属性、用户-项目评分矩阵和评分时间计算用户显隐兴趣相似度；4)融合用户评分相似度和用户显隐兴趣相似度得到用户相似度；5)寻找目标用户最近邻并对未产生行为项目进行评分预测，把评分高的项目推荐给用户。该算法的流程如算法1所示。

算法1 融合用户评分与显隐兴趣相似度的协同过滤算法

输入：用户-项目评分矩阵$\mathbf{R}_{m\times n}$，用户基本属性$\mathbf{U}_{m\times l}$，项目显性属性矩阵$\mathbf{E}_{n\times k}$，目标用户u

输出：用户u的前n个项目

开始：

1. for 每个项目 i:
2. 根据表 2 计算出 S_P^i, S_N^i;
3. end for;
4. for 每个用户 v:
5. 根据式(5)－式(12)计算出用户评分相似度
6. $sim(u,v)^R$;
7. end for;
8. for 每个项目 i:
9. for 每个属性 a:
10. for 属性 a 中每个取值 j:
11. 根据式(16)计算出项目 i 的隐性属性
12. I_{aj}^i;
13. end for;
14. end for;
15. end for;
16. 构建项目隐性属性矩阵 $\mathbf{I}_{n\times k}$;
17. for 每个用户 v:
18. 根据式(17)－式(21)计算用户兴趣相似度;
19. 根据式(22)计算用户最终相似度;
20. end for;
21. 根据用户最终相似度寻找 k 个最近邻
22. for 每个项目 i:
23. 根据式(23)计算用户 u 对项目 i 的预测评分;
24. end for;
25. 输出评分最高的前 n 个项目;
结束。

5 实验结果和分析

为了能够验证融合用户评分与显隐兴趣相似度的协同过滤算法的有效性,本文在开源的 Movielens 数据集上,将其与传统的 3 种用户相似度算法、文献[10]和文献[12]提出的算法进行比较。实验结果表明,所提出的新的相似度算法的准确性更好。该实验环境为:python,3.70 GHz CPU,16 GB 内存,Windows 10 操作系统。

5.1 数据集

实验采用美国 Minnesota 大学的 GroupLens 研究小组创建并维护的 Movielens 数据集,该数据集包含用户对电影的评分信息、电影的类别属性信息(显性属性)以及用户的基本属性信息。其中用户有 943 个、电影有 1 682 部、评分信息有 10 万条,评分信息有用户 ID、项目 ID、评分、评分时间,评分范围是[1,5];电影属性信息包括电影所属的类别,包括如喜剧、儿童剧等 19 个类别;用户属性信息包括性别、年龄、职业、地址。本文实验将数据集分成 80%的训练集和 20%的测试集,训练集用于计算用户之间的相似度,然后对测试集中的电影预测评分。由第 3.2.3 节中项目隐性属性的构建方法可知,本实验根据用户基本属性中的年龄、性别、职业构建项目隐性属性,然后对用户的年龄、职业进行离散化处理,将年龄分为 6 组,分组信息即[0,22],[23,27],[28,31],[32,39],[40,48],[49,∞];将职业分为 5 组,分组信息即[doctor,homemaker,none,salesman,lawyer,retired,healthcare,entertainment,marketing,technician,artist],[scientist,executive,writer,librarian],[programmer,engineer,administrator],[educator],[other,student]。

5.2 评价指标

实验使用经典评价指标平均绝对误差 MAE(Mean Absolute Error)和均方根误差 RMSE(Root Mean Square Error)来评价所提算法的质量。平均绝对误差如式(24)所示,其通过计算预测评分值和真实评分值的差异来度量预测的准确度。

$$MAE=\frac{\sum_{u,i\in test}|r_{ui}-r'_{ui}|}{N} \tag{24}$$

其中,r_{ui} 表示用户 u 对项目 i 的真实评分,r'_{ui} 表示预测评分,N 表示预测评分的个数。

为了能更好地比较算法的效果,采用均方根误差进行比较,其计算公式如式(25)所示:

$$RMSE=\sqrt{\frac{\sum_{u,i\in test}|r_{ui}-r'_{ui}|^2}{N}} \tag{25}$$

其中,r_{ui} 表示用户 u 对项目 i 的真实评分,r'_{ui} 表示预测评分,N 表示预测评分的个数。

5.3 参数选择

式(21)中的 β 和式(22)中的 α 分别表示用户兴趣相似度中显性兴趣相似度和隐性兴趣相似度的比重和用户相似度中用户评分相似度和用户显隐兴趣相似度的比重。它们的范围均是[0,1],为了选择最优的参数,本实验设置不同的 α 与 β 值进行遍历,寻找 MAE 最低时的参数,用其作为后续实验的参数。

如图 1 所示,横坐标表示参数 α 的取值,每一条线表示一个参数 β 的取值,纵坐标表示参数取不同值时的 MAE。由图 1 可知,当只使用用户评分或用户兴趣相似度时(即当 $\alpha=0$ 或 $\alpha=1$ 时),MAE 值都比较高,而当两者进行融合后,MAE 值便会下降;当 MAE 值为最小值时有多个参数对可以选择,因此本实验随机选择 $\alpha=0.4$,$\beta=0.3$ 作为实验参数进行后续实验。

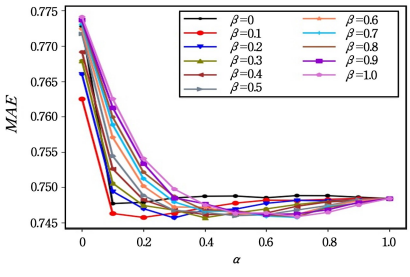

图 1 参数选择参考图

Fig. 1 Parameter selection reference figure

5.4 实验结果

确定 α 与 β 的参数取值后,将本文方法与传统的相似度算法在评价指标 MAE 上进行比较。

如图 2 所示,横坐标为相似用户数 K 的取值,纵坐标表示在不同 K 值下不同算法的 MAE 值。由图 2 可知,随着近邻数 K 的增加,各个算法的 MAE 值呈下降趋势;本文方法相比其他相似度方法,其整体 MAE 都更小;COS、ACOS 和文献[10]的方法相差不大;PCC 和文献[12]的方法的 MAE 整体都很大;当 K 取[20,60]之间时,本文方法的 MAE 值综合表

现更好。*MAE* 随着 K 的变化而变化是因为寻找的相似用户不够准确，对预测结果的准确性有所干扰；各个算法最终都需要将相似度映射到区间[0,1]中，不同的映射结果导致了算法预测结果的差异。

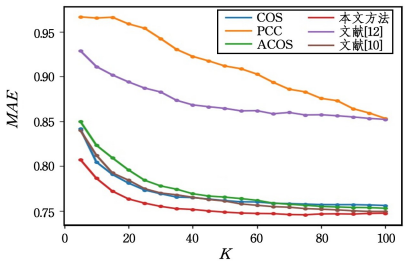

图 2 *MAE* 结果

Fig. 2 *MAE* results

当对测试集中的评分进行预测时，虽然会通过 K 个最相似用户对目标项目的评分进行预测，但是当 K 个最相似用户都没有对目标项目有过评分行为时，便无法进行预测，因此，如果算法能够预测评分的项目更多，就表明算法寻找到的相似用户更准确。

如图 3 所示，横坐标表示相似用户数 K 的取值，纵坐标表示在不同 K 值下，不同算法能够预测出评分的项目个数占总预测项目个数的比例。由图 3 可知，随着相似用户数 K 的增加，各个算法能够预测的评分个数也在不断增加；并且本文方法能够预测评分的项目个数比例整体较高，当 $K<20$ 时，本文方法的表现最好，这也从侧面反映出根据本文方法找到的前 20 个相似用户更准确。

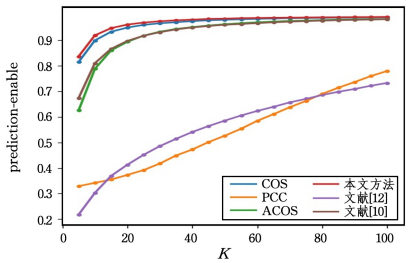

图 3 可预测评分项目比例图

Fig. 3 Prediction-enable item proportion figure

如图 4 所示，横坐标为相似用户数 K 的取值，纵坐标表示在不同 K 值下不同算法的 *RMSE* 值。由图 4 可知，随着近邻数的增加，各个算法的 *RMSE* 都在下降，而本文提出的新算法更早稳定且整体偏低，推荐准确度更优。

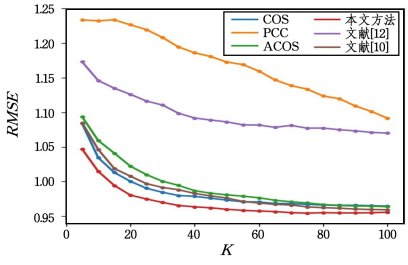

图 4 *RMSE* 结果

Fig. 4 *RMSE* results

算法运行时，需要根据训练集计算出每个用户之间的相似度，表 8 列出了各个算法计算相似度所需的时间，其中，评分相似度表示根据用户-项目评分矩阵计算相似度所需要的时间，其他相似度表示根据用户基本属性、项目类别属性等数据计算相似度所需要的时间。由表 8 可知，计算相似度时考虑的信息越多，花费的时间也就越多；传统的相似度计算方法相对简单，因此它们的计算时间较少；文献[10]引入了项目类别属性计算用户相似度，因此它的计算时间略高于传统方法；文献[12]引入了信息熵以及评分时间计算相似度，因此计算时间相对前 4 种方法偏高；本文方法在以上方法考虑因素的基础上引入了项目类别属性计算相似度，因此计算时间相对偏高。由表 8 可知，本文所提算法更加适用于小数据集，在大数据集情况下，可以考虑使用分布式计算以提高算法性能。

表 8 运行时间表

Table 8 Runing time

（单位：s）

	评分相似度时间	其他相似度时间	总时间
COS	857.985	0	857.985
PCC	1 327.253	0	1 327.253
ACOS	1 092.805	0	1 092.805
文献[10]	1 453.097	185.785	1 638.882
文献[12]	2 915.942	184.382	3 100.324
本文方法	3 612.174	367.365	3 979.539

综上所述，本文算法在 MAE 和 RMSE 评价指标上的表现相比其他算法更优秀、更稳定，而且寻找到的相似用户更准确，但其在运行时间上略微高于其他算法，因此本文所提算法是一个针对于小规模数据集的有效可行的协同过滤算法。

结束语 由于数据的稀疏性问题，传统协同过滤算法中仅使用用户-项目评分数据来计算相似度的方法会导致推荐质量不高。本文充分利用易获取数据，提出融合用户评分与显隐兴趣相似度的协同过滤算法，将用户的相似度分为用户评分相似度和用户显隐兴趣相似度两个部分。其中，用户评分相似度再分为 3 个子相似度，即奇异相似度、调整的杰卡德相似度、评分差异相似度；用户显隐兴趣相似度分为两个子相似度，即用户显性兴趣相似度和用户隐性兴趣相似度。实验结果表明，提出的新的相似度计算方法不仅能够准确地找到相似用户，而且还能够有效提升推荐质量。

由于算法在时间效率上略显不足，后续工作将考虑优化算法步骤，以提高时间性能；考虑到将项目的发行时间与用户的评分时间相结合也可能反映出用户的兴趣，因此后续工作也将考虑引入项目发行时间，对相似度计算进行改进，以便更好地提升推荐质量。

参 考 文 献

[1] LI M M, WANG L C. A Survey on Personalized News Recommendation Technology[J]. IEEE Access, 2019, 7(99): 145861-145879.

[2] JALILI M, AHMADIAN S, IZADI M, et al. Evaluating Collaborative Filtering Recommender Algorithms: A Survey[J]. IEEE Access, 2018, 6: 74003-74024.

[3] NATARAJAN S, VAIRAVASUNDARAM S, NATARAJAN S, et al. Resolving data sparsity and cold start problem in collaborative filtering recommender system using Linked Open Data

[J]. Expert System with Applications, 2020, 149: 113248.
[4] WU C H, WU F Z, QI T, et al. Reviews Meet Graphs. Enhancing User and Item Representations for Recommendation with Hierarchical Attentive Graph Neural Network[C]// EMNLP/IJCNLP. 2019: 4883-4892.
[5] SHI W C, WANG L J, QIN J W. User Embedding for Rating Prediction in SVD++-Based Collaborative Filtering[J]. Symmetry, 2020, 12: 121.
[6] GUO X, ZHU J H. Deep Neural Network Recommendation Model Based on User Vectorization Representation and Attention Mechanism[J]. Computer Science, 2019, 46(8): 111-115.
[7] LIU H F, HU Z, AHMAD U M, et al. A new user similarity model to improve the accuracy of collaborative filtering[J]. Knowledge Based System, 2014, 56(Jan.): 156-166.
[8] TAN Z H, HE L L. An Efficient Similarity Measure for User-Based Collaborative Filtering Recommender Systems Inspired by the Physical Resonance Principle[J]. IEEE Access, 2017, 5: 27211-27228.
[9] JIN Q B, ZHANG Y, CAI W, et al. A New Similarity Computing Model of Collaborative Filtering[J]. IEEE Access, 2020, 8: 17594-17604.
[10] WANG Y C, LIU Z. Collaborative Filtering Algorithm Based on User's Preference for Items and Attribut [J]. Computer Science, 2018, 45(S2): 422-426.
[11] LI Z L, HUANG M X, ZHANG Y. A Collaborative Filtering Algorithm of Calculating Similarity Based on Item Rating and Attributes[C]// Web Iinformation Systems and Applications. 2017: 215-218.
[12] WEI H J, DAI D H. Collaboration Filtering Recommendation Algorithm Based on Ratings Difference and Interest Similarity [J]. Computer Science, 2018, 45(S1): 411-414, 435.
[13] ZHANG L, ZHANG Z J, HE J F, et al. UR: A User-Based Collaborative Filtering Recommendation System Based on Trust Mechanism and Time Weighting[C]// 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS). 2020: 69-76.

WU Jian-xin, born in 1995, postgraduate. His main research interests include data mining and artificial intelligence.

ZHANG Zhi-hong, born in 1965, Ph.D, professor, is a member of China Computer Federation. His main research interests include block chain and financial big data.