针对稀疏评分矩阵的改进协同过滤推荐算法

周子亮，吴为民
（北京交通大学计算机与信息技术学院，北京 100044）

摘要：协同过滤推荐策略是推荐系统中应用最成功的个性化推荐策略，然而，数据匮乏一直是推荐系统发展所面临的重大挑战。本文针对这一问题，从三个方面来应对用户评分矩阵的稀疏性问题：首先，调整相似度度量方法，其次，结合基于用户的协同过滤和基于项目的协同过滤，生成了新的预测模型，最后，提出了一种迭代预测算法。论文最后使用 MovieLens 数据集对算法进行评价，并对比其他传统算法。实验结果表明改进算法能够更好的处理用户评分矩阵的稀疏性问题。

关键词：协同过滤；相似度；预测模型；迭代预测算法
中图分类号：TP391.1

An Improvement Collaborative Filtering Algorithm for Sparsity Ratings Matrix

Zhou Ziliang, Wu Weimin
(Computer and Information Technology School, Beijing Jiaotong University, Beijing 100044)

Abstract: As the most successful personalized recommendation algorithm, collaborative filtering is widely used in recommendation system. However, the lack of data has been a huge challenge. To deal with the sparsity problem, we provide an algorithm with three improved methods. On the one hands we adjusted the similarity metrics. On the other hand, we combine User-based collaborative filtering with Item-based collaborative filtering. Finally, we provide an iterative algorithm to predict the rate given by the active user. In the experiment section, we evaluated our new algorithm using the MoiveLens dataset. The results suggest that the new algorithm can better handle the user rating matrix sparsity problem.

Keywords: collaboration filtering; similarity; prediction model; iterative algorithm

0 引言

随着信息技术和互联网的发展，人们逐渐从信息匮乏的时代走进了信息过载的时代，这个时代的特点是海量数据，用户很难在这些数据中找到自己需要的信息，因此推荐系统应运而生。目前推荐系统已经被广泛应用于电子商务、视频网站、新闻网站等场合。

协同过滤推荐策略是推荐系统中应用最成功的个性化推荐策略，协同过滤算法依靠的基础是用户和项目之间的评分交互信息[1]，一般采用用户-项目的评分矩阵来表示，然而随着互联网的发展，网络用户和网络上的信息迅速增多，这种交互信息所占的比例越来越小，也就是数据稀疏性，一般来讲，一个网站上用户对项目的评分不会超过矩阵大小的 1%，导致推荐系统的精准度下降。

1 相关研究

协同过滤推荐算法一般包括基于内存的协同过滤和基于模型的协同过滤，基于内存的协同过滤又包括基于用户和基于项目两种。基于用户的协同过滤根据用户之间评分向量的相似

作者简介：周子亮，(1985-)，男，主要研究方向：信息检索。
通信联系人：吴为民，(1966-)，男，副教授，主要研究方向：数字系统设计自动化，嵌入式系统，形式验证。E-mail: wmwu@bjtu.edu.cn

度来寻找用户的最近邻用户，根据最近邻用户的评分来为目标用户推荐相应的项目；同样地，基于项目的协同过滤计算项目之间的相似度，并依此得出项目得邻居项目，然后根据邻居项目的评分用户来决定是否应该将该项目推荐给相应的用户。

1.1 数据稀疏性

随着数据稀疏性的增加，用户之间的相似度计算会变异常困难，得出的邻居节点甚至有很少的共同评分项目。导致给目标用户推荐的项目并不是该用户所偏好的项目。

下面我们分析 Movielens 数据集中数据稀疏性给最近邻节点的查找带来的影响，Movielens 数据集是业界非常著名的数据集，源自 Minnesota 大学的 GroupLens 研究小组，它包含了 943 个用户对 1682 部电影的 100000 个评分项，其中用户编号为 1~943 用 u_m 表示，电影编号为 1~1682，用 i_n 表示，用户评分值用 1 到 5 的整数表示。我们用稀疏度表示数据稀疏性程度[2]，稀疏度越高则稀疏性越高，Movielens 数据集的稀疏度为：1-(100000/(9431682))=93.69%。

表 1.电影 i_{20} 的前 20 个邻居节点 u_2

Tab. 1 The nearest 20 neighbors of movie 20

排序	电影编号	Pearson 相似度	评分
1	828	1	0
2	1008	0.94809085	0
3	298	0.943721	3
4	291	0.94022971	0
5	1198	0.89375615	0
6	764	0.8896693	0
7	886	0.88013458	0
8	574	0.87775487	0
9	978	0.87065923	0
10	323	0.8564924	2
11	875	0.84797043	0
12	341	0.8396579	4
13	111	0.82710308	0
14	1214	0.81995988	0
15	1016	0.819959598	0
16	1032	0.81522197	0
17	968	0.80054921	0
18	242	0.79666167	0
19	515	0.79655576	0
20	869	0.79003358	0

表 1 中评分栏表示用户 u_2 对电影的评分，可以看出在 i_{20} 的前 20 个邻居中，只有 3 个电影被用户 u_2 评价过，这样预测用户 u_2 对电影 i_{20} 的评分就变的非常不可靠，可以通过寻找被用户 u_2 评过分的与电影 i_{20} 相似度最高的 20 个最近邻电影：

表 2 用户 u_2 对电影 i_{20} 的评分中前 20 个邻居节点

Tab.2 The nearest 20 neighbors of movie 20 rated by user 2

排序	电影编号	Pearson 相似度	评分
3	298	0.943721	3
10	323	0.85649234	2
12	341	0.8396579	4
31	329	0.73737031	2
70	330	0.62249732	4
73	325	0.61980271	2
81	354	0.59999985	3
121	342	0.53304631	3
159	333	0.47130455	3
175	328	0.45808184	4
187	324	0.44850534	1
191	263	0.438350598	2
195	321	0.43201941	3
204	353	0.4199127	3
205	344	0.41922736	3
230	336	0.39906901	1
311	332	0.33391178	2
329	318	0.32190725	2
331	270	0.32106668	3
450	180	0.231359006	4

由表 1 和表 2 可以看出，预测用户 u_2 对电影 i_{20} 的评分，使用第一种传统的邻居节点选取方法的话会导致用户 u_2 对电影 i_{20} 的近邻电影评分的稀疏，从而使得预测得到的分值较小。

1.2 相似度计算方法

1.2.1 传统的相似度度量方法

传统的相似度度量方法主要有以下几种[3]：余弦(Cosine)相似度、修正的余弦(Adjusted Cosine)相似度、皮尔逊(Pearson)相关系数。下面我们通过定义两个用户 u_p 和 u_q 的相似度，来给出三种相似度的定义，设 u_p 和 u_q 对应的评分向量为 $\vec{p}$ 和 $\vec{q}$，用户对项目的评分矩阵为 $R \in \mathbb{R}^{M\times N}$ (M 个用户 N 个项目)，r_{pk} 表示 R 中第 p 行 k 列元素，也就是用户 u_p 对项目 i_k 的评分值。

余弦相似度采用向量的余弦夹角度量两个向量之间的相似性，式(1)给出了余弦相似度的公式：

$$sim(u_p,u_q)=\cos(\vec{p},\vec{q})=\frac{\vec{p}\bullet\vec{q}}{\|\vec{p}\|\times\|\vec{q}\|}=\frac{\sum_{k=1}^{N}r_{pk}r_{qk}}{\sqrt{\sum_{k=1}^{N}r_{pk}^2}\sqrt{\sum_{k=1}^{N}r_{qk}}} \tag{1}$$

余弦相似性没有考虑两个用户自身的评价尺度问题，比如有的用户非常宽容，对不喜欢的电影也评一个不错的分数，而有的用户比较苛刻，对喜欢的电影也不会给太高的分数。考虑到这些，我们使用修正的余弦相似度：

$$sim(u_p,u_q)=\frac{\sum_{k=1}^{N}(r_{pk}-\overline{r_p})(r_{qk}-\overline{r_q})}{\sqrt{\sum_{k=1}^{N}(r_{pk}-\overline{r_p})^2}\times\sqrt{\sum_{k=1}^{N}(r_{qk}-\overline{r_q})^2}} \tag{2}$$

其中 $\overline{r_p}$ 和 $\overline{r_q}$ 表示用户 u_p 和 u_q 在评分矩阵上所有评分的平均值。

最后，我们来讨论皮尔逊相关系数，皮尔逊相关系数在打分制的体系中应用比较广泛，

但是只考虑了两个空间向量都包含的维度进行计算：

$$sim(u_p,u_q)=\frac{\sum_{k=1}^{N}r_{pk}r_{qk}-N\overline{r_p}\times\overline{r_q}}{(N-1)s_p s_q}$$

$$=\frac{N\sum_{k=1}^{N}r_{pk}r_{qk}-\sum_{k=1}^{N}r_{pk}\sum_{k=1}^{N}r_{qk}}{\sqrt{N\sum_{k=1}^{N}r_{pk}^2-(\sum_{k=1}^{N}r_{pk})^2}\sqrt{N\sum_{k=1}^{N}r_{qk}^2-(\sum_{k=1}^{N}r_{qk})^2}} \quad (3)$$

其中 s_p、s_q 是向量 $\vec{p}$ 和 $\vec{q}$ 的标准方差。

1.2.2 改进的相似性度量方法

对于一些热门项目，会有很多用户对其进行评价，由于数据的稀疏性，使得这些用户很容易成为邻居用户，热门项目不能够反映用户的兴趣。比如，如果两个用户都买过《英汉词典》，并不能说他们的兴趣相似，但是如果两个用户都买过《机器学习导论》，则可以认为他们的兴趣是相似的。也就是说，两个用户对冷门物品采取过同样的行为更能说明他们兴趣的相似，因此，John S. Breese 提出了一种新的相似度公式[4]：

$$sim(u_p,u_q)=\frac{\sum_{i_k\in N(u_p)\cap N(u_q)}\frac{1}{\log(1+|N(i_k)|)}}{\sqrt{|N(u_p)||N(u_q)|}} \quad (4)$$

其中 $N(p)$、$N(q)$ 表示用户 u_p、u_q 评价过的项目的集合，$N(k)$ 表示项目 i_k 被评价过的用户的集合。不难看出该公式通过 $1/\log(1+|N(k)|)$ 惩罚了用户 u_p 和 u_q 共同兴趣列表中热门物品对他们相似度的影响。

进一步，我们增加一个权重因子来应对数据的稀疏性，权重因子使用两个用户之间共同评分项目占总评分项目的比重，这样就减少了点评过很多项目的用户的比重。最终的相似度形如：

$$S(u_p,u_q)=(1-\mu)+\mu\frac{|N(u_p)\cap N(u_q)|}{|N(u_p)\cup N(u_q)|}sim(u_p,u_q) \quad (5)$$

其中 μ 是一个可以手动设置的权重指数，且 $\mu\in(0,1]$。将公式(4)代入公式(5)中即可得到最终的改进的相似度公式。

1.3 邻居节点的产生

邻居节点的产生一般有两种方法，一种是 K 近邻（KNN，k-Nearest Neighbor）算法，一种是 K 阈值算法，图(1)给出了两种算法是示意图。

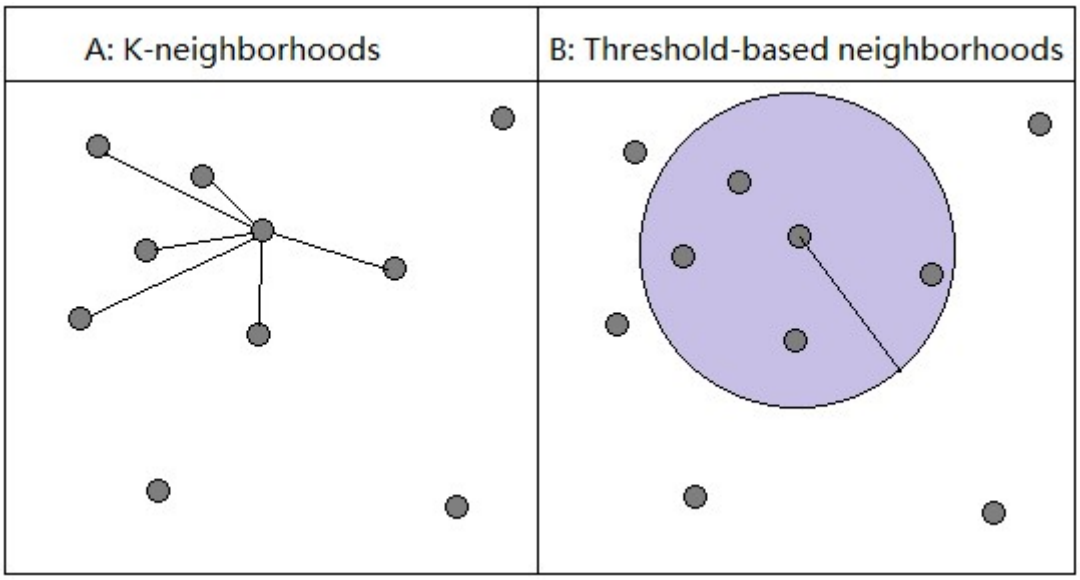

图 1　KNN 和 K 阈值两种产生邻居节点策略示意图
Fig. 1　The KNN algorithm and the K threshold algorithm

KNN 算法通过寻找目标节点的最近的 N 个节点作为它的邻居，缺点是它产生的邻居有可能相似度比较小，从而会影响产生推荐的精确性。K 阈值算法通过限定它的阈值来寻找目标节点的邻居节点，所有与目标节点的相似度小于这个阈值的节点都是该节点的邻居节点，它的缺点是有可能有些节点找不到邻居节点。

这里我们形式化介绍 K 阈值算法：给定用户阈值 α 、项目阈值 β，用户 u_m 的最近邻用户集合可以表示成：

$$N(u_m)=\{u_a \mid S(u_m,u_a)>\alpha, u_a \neq u_m\} \tag{6}$$

项目 i_n 的最近邻用户集合可以表示成：

$$N(i_n)=\{i_b \mid S(i_n,i_b)>\beta, i_b \neq i_n\} \tag{7}$$

1.4 推荐结果

在得出用户的邻居节点后，我们可以由此产生推荐结果，推荐结果的产生就是对评分矩阵 R 中的空值进行填充的过程，是通过预测活跃用户对项目的评分来实现的，根据协同过滤的分类，这里我们介绍使用基于用户的协同过滤和基于项目的协同过滤两种算法产生结果的方式。

1.4.1 基于用户的协同过滤推荐产生的结果

对于目标用户 u_a，如果该用户在评分矩阵 R 中对项目 i_n 没有评分值，那么我们可以根据以下公式来预测该评分值：

$$\hat{r}_{an}=\overline{u_a}+\frac{\sum_{u_m \in N(u_a)} S(u_a,u_m)(r_{mn}-\bar{r}_m)}{\sum_{u_m \in N(u_a)} |S(u_a,u_m)|} \tag{8}$$

其中 $\bar{r}_a$ 和 $\bar{r}_m$ 分别表示用户 u_a 和用户 u_m 对所有项目评分的平均值，$S(u_a,u_m)$ 表示用户 u_a 和用户 u_m 的相似度。 $N(u_m)$ 表示用户 u_m 的邻居集合。

1.4.2 基于项目的协同过滤推荐产生的结果

同样，对于目标用户u_a，如果该用户在评分矩阵R中对项目i_n没有评分值，那么我们可以根据以下公式来预测该评分值：

$$\hat{r}_{an}=\bar{r}_n+\frac{\sum_{i_k\in N(i_n)}S(i_n,i_k)\times(r_{ak}-\bar{r}_k)}{\sum_{i_k\in N(i_n)}|S(i_n,i_k)|} \tag{9}$$

其中$\bar{r}_n$和$\bar{r}_k$分别表示项目i_n和项目i_k所得评分的平均值，$S(i_n,i_k)$表示项目i_n和项目i_k的相似度。$N(i_n)$表示项目i_n的邻居集合。

2 改进推荐策略

本节对以上算法继续进行改进，以更好的应对数据稀疏性问题，下面我们通过两个方面来对算法进行改进。

2.1 对推荐模型进行改进

对于公式(8)和公式(9)，当数据非常稀疏时，有可能导致用户或者项目的邻居集合为空，传统的推荐算法是把该预测值设置成用户所有评分的平均值，但这显然不能代表用户的兴趣度，所以，我们提出一种结合两种推荐策略的方法。将基于用户的协同过滤和基于项目的协同过滤组合起来，则预测用户u_a对项目i_n的评分为：

$$\hat{r}_{an}=\sigma\left[\bar{u}_a+\frac{\sum_{u_m\in N(u_a)}S(u_a,u_m)(r_{mn}-\bar{r}_m)}{\sum_{u_m\in N(u_a)}|S(u_a,u_m)|}\right]+$$

$$(1-\sigma)\left[\bar{r}_n+\frac{\sum_{i_k\in N(i_n)}S(i_n,i_k)\times(r_{ak}-\bar{r}_k)}{\sum_{i_k\in N(i_n)}|S(i_n,i_k)|}\right] \tag{10}$$

其中σ表示用户和项目的影响因素的比例调节因子。其余符号同公式(8)公式(9)相同。

调节因子σ用来调节矩阵的稀疏程度，相对而言，如果基于用户协同过滤推荐算法中目标用户的邻居集合比较少时，可以适当地增加基于项目的协同过滤算法所占的比重，反之亦然。当目标用户的邻居集合不为空，且项目的邻居集合为空时，则只依靠基于项目的协同过滤策略，所以σ为1，相反，当用户邻居集合为空，项目的邻居集合不为空时，则依赖于基于用户的协同过滤策略，因此σ为0，当二者都为空时，说明该点为离群点，无法为此预测评分，可以适当采用基于内容的过滤等其他方法进行推荐。因此定义σ的取值形式如下：

$$\sigma=\begin{cases}1 & N(u_a)\neq\varnothing;N(i_n)=\varnothing\\ \dfrac{\eta_1}{\eta_1+\eta_2} & N(u_a)\neq\varnothing;N(i_n)\neq\varnothing\\ 0 & N(u_a)=\varnothing;N(i_n)\neq\varnothing\end{cases} \tag{11}$$

2.2 迭代预测算法

通过协同过滤的过程，在预测用户对项目的评分时，有可能使得用户评分矩阵变得更加稀疏，针对这一问题，我们通过人为的在空白处加入数据来减小稀疏度，加入的值通过优化得到一个最优值，求最优化的过程是一个递归迭代的过程，下面我们加入递归因子θ_{mn}，则

评分公式变为：

$$\hat{r}_{an}=\sigma\left[\bar{u}_a+\frac{\sum_{u_m\in N(u_a)}\theta_{mn}S(u_a,u_m)(r_{mn}-\bar{r}_m)}{\sum_{u_m\in N(u_a)}\theta_{mn}\left|S(u_a,u_m)\right|}\right]+$$

$$(1-\sigma)\left[\bar{r}_n+\frac{\sum_{i_k\in N(i_n)}\theta_{ak}S(i_n,i_k)\times(r_{ak}-\bar{r}_k)}{\sum_{i_k\in N(i_n)}\varphi_{ak}\left|S(i_n,i_k)\right|}\right] \tag{12}$$

其中：

$$\theta_{mn}=\begin{cases}1 & r_{an}\neq 0\\ \varepsilon & r_{an}=0\end{cases} \tag{13}$$

由公式(12)、(13)可知，当评分矩阵中 r_{ak} 项不为空时，预测评分公式按照原公式进行，当评分矩阵中 r_{ak} 为空时，则通过递归因子 θ_{mn} 和 φ_{ak} 赋予该项一个系数，也就是使得 $r_{ak}-\bar{r}_k$ 不再以默认值平均值处理。我们通过迭代的方法对初始化的用户评分进行求解，规定迭代阈值 Σ，迭代次数超过 Σ 时就停止迭代，算法如下：

算法 1 针对数据稀疏性改进的协同过滤推荐算法

Tab. 1 An Improvement Collaborative Filtering Algorithm for Sparsity Ratings Matrix

输入：用户评分矩阵 R；

目标用户 u_a 和目标项目 i_n 的编号；

迭代次数阈值 Σ，线性组合参数 μ；

输出：目标用户 u_a 对目标项目 i_n 的预测评分值。

步骤 1：利用公式(5)计算目标用户 u_a 和其他用户的相似度以及活跃项目 i_n 和其他项目的相似度；

步骤 2：通过公式(6)和公式(7)对求取目标用户和目标项目的邻居集合。

步骤 3：对于目标用户 u_a 和目标项目 i_n 的邻居集合 $N(u_m)$ 和 $N(i_n)$ 中的所有项目：

If（R 中没有用户 u_a 对项目 i_n 的评分，且当前迭代次数小于阈值 Σ）：

预测用户 u_a 对项目 i_n 的评分；

Else： 转到步骤 5；

步骤 4：If（迭代次数小于阈值 Σ）：

将预测的用户 u_a 对项目 i_n 的评分 $\hat{r}_{an}$ 代入 θ_{an}；

迭代次数加 1，转到步骤 3；

步骤 5：根据公式(12)和公式(13)求得目标用户对目标项目的预测得分。

3 实验分析

3.1 数据集

实验采用著名的 Movielens 数据集进行验证，该数据集已经在 1.1 节进行过介绍。

3.2 评价指标

本文主要推荐精确度进行验证，采用的是平均绝对误差(MAE)，MAE 用来衡量推荐系统预测的评分值与用户实际评分值之间的偏差，是应用最为广泛的一种评价指标[5,6,7]。

定义 1 设目标用户的预测评分集合为 $\{\hat{r}_1,\hat{r}_2,...,\hat{r}_t\}$，而实际评分集合为 $\{r_t,r_t,...,r_t\}$，则平均绝对误差 MAE 为：

$$MAE = \frac{\sum_{j=1}^{t} \left| \hat{r}_j - r_j \right|}{t} \tag{14}$$

3.3 实验运行及分析

为了体现该算法的有效性，我们将该算法与基于用户的协同过滤算法(UserCF)、基于项目的协同过滤推荐系统[8](ItmeCF)进行对比，方便起见，我们把该算法叫做 AdvancedCF。

首先对线性组合参数 μ 进行试验，如下图：

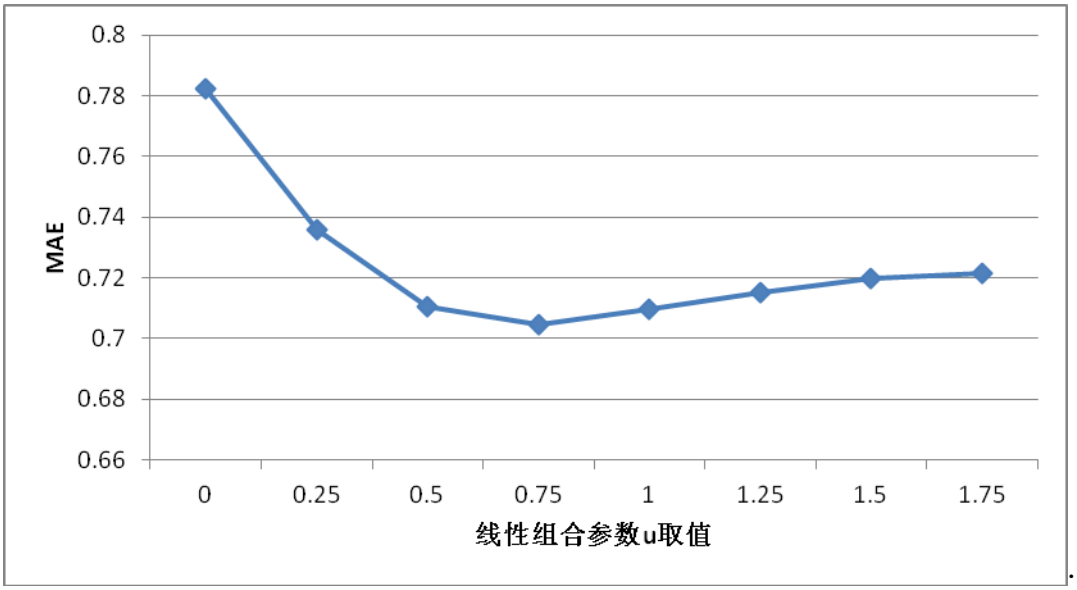

图 2：线性组合参数 μ 对算法的影响

Fig.2 The influence of the parameter μ

由图 2 可以看出，在组合参数 μ 取值为 0.75 附近时，MAE 取得最小值，也就是系统达到最大的精确值。

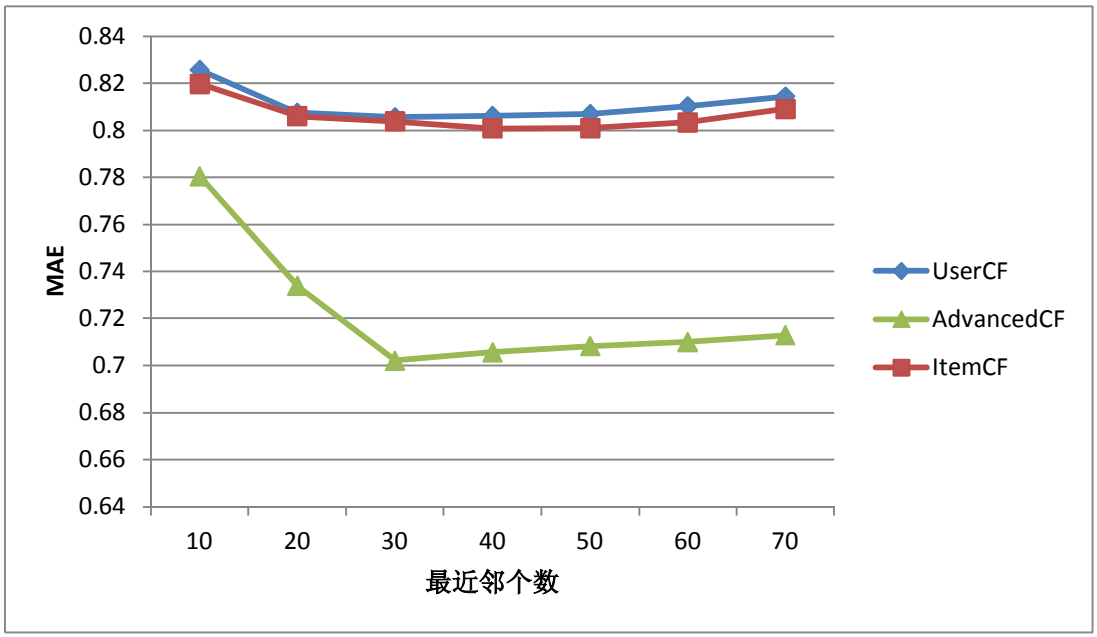

图 3 与系统过滤推荐算法比较

Fig. 3 Compare with the traditional CF Algorithm

通过图 3 可以清楚地看到，我们的推荐算法有较高的精确度，能够有效地处理高维稀疏数据。当最近邻个数达到 30 左右时，算法可以达到一个较好的精确度，随着最近邻个数的继续增加，精准度反而下降，我们推测是应为邻居个数增多，会引入很多相似度不高的用户和项目，导致带来目标用户不感兴趣的物品。

4 总结

推荐系统是新兴的重点研究领域和应用技术，随着互联网的发展，用户数目和项目数目急剧增加，导致了数据稀疏的问题，针对这一问题，本文提出一种新的协同过滤算法，实验证明，该方法能够有效地应对数据稀疏问题，具有较高的实用价值。

[参考文献] (References)

[1] U. Shardanand and P. Maes, Social Information Filtering: Algorithms for Automating 'Word of Mouth', Proc. Conf. Human Factors in Computing Systems, 1995: 210-217.

[2] 曾春，邢春晓，周立柱等.个性化服务技术综述[J].软件学报，2002,13(10)：1952-1961

[3] 李雪，左万利，赫枫龄等.传统 Item-Based 协同过滤推荐算法改进[J].计算机研究与发展，2009,36(4)：614-618,638.

[4] John S. Breese, David Heckerman, Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering[J].Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence(UAI'98),Page 43-52.

[5] Breese, J. S., Heckerman, D., and Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In Proceedings of the 14th Conference onUncertainty in Artificial Intelligence, 1998,43-52.

[6] 侯翠琴，焦李成，张文革等.一种压缩稀疏矩阵的协同过滤算法[J].西安电子科技大学学报(自然科学版)，2009,36(4):614-618.

[7] Jonathan L. Herlocker , Joseph A. Konstan , Al Borchers , John Riedl, An algorithmic framework for performing collaborative filtering, Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, p.230-237, August 15-19, 1999, Berkeley, California, United States. 1999, 230-237

[8] Jun Wang, Arjen P. de Vries, Marcel J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion[C].Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. Washington DC, USA, 2006:501-508.