第35卷 第23期 计 算 机 工 程 2009年12月
Vol.35 No.23 Computer Engineering December 2009
·人工智能及识别技术· 文章编号：1000—3428(2009)23—0178—03 文献标识码：A 中图分类号：TP301.6

# 基于粒子群优化的项聚类推荐算法

熊忠阳，张凤娟，张玉芳
(重庆大学计算机学院，重庆 400030)

**摘 要：**针对传统推荐算法的数据稀疏性问题和推荐准确性问题，提出基于粒子群优化的项聚类推荐算法。采用粒子群优化算法产生聚类中心，在此基础上搜索目标项目的最近邻居，并产生推荐，从而提高了传统聚类算法的推荐准确性及响应速度。实验表明改进的项聚类协同过滤算法能有效提高推荐精度。

**关键词：**粒子群优化；项聚类；协同过滤；推荐算法

# Item Clustering Recommendation Algorithm Based on Particle Swarm Optimization

XIONG Zhong-yang, ZHANG Feng-juan, ZHANG Yu-fang
(School of Computer, Chongqing University, Chongqing 400030)

**【Abstract】**Aiming at the problems that the data are sparse and the results are not accurate in traditional recommendation algorithms, this paper proposes an item clustering recommendation algorithm based on Particle Swarm Optimization(PSO) algorithm. It uses PSO to engender the cluster centers, calculates the similarity between target item and cluster centers to search the nearest neighbors of target item, and gains a recommendation, so that it improves the accuracy and the real-time performance. Experimental results indicate that the algorithm can effectively improve the accuracy of the recommendation system.

**【Key words】**Particle Swarm Optimization(PSO); item clustering; collaborative filtering; recommendation algorithm

## 1 概述

协同过滤是最早最成功的推荐技术[1]，主要是利用目标用户对产品的历史评价与其他用户相匹配，预测用户对未评价产品的评价，从而产生推荐。虽然协同过滤技术得到了广泛的应用，但随着商务规模的扩大、用户数量与商品数量的增加，也暴露出一些缺点，主要体现在数据的稀疏性问题、系统的可扩展性问题、推荐的实时性、推荐的精度等。

针对电子商务推荐系统存在的问题，本文提出基于粒子群优化(Particle Swarm Optimization, PSO)的项聚类推荐算法。用粒子群优化的聚类算法将用户评分相似的项目放入同一个聚类中，再计算目标项目与每个聚类中心的相似性，以相似性最高的几个聚类作为目标项目的查询空间进行搜索，找出相似性最高的项目作为最近邻居，根据这些最近邻居的评分预测用户对目标项目的评分并产生最终的推荐列表。通过聚类减小了搜索空间，提高了推荐的响应时间，利用PSO算法产生的聚类具有较高的精度，从而提高了推荐的精度，实验结果也表明本文方法具有较好的推荐效果。

## 2 基于项聚类的协同过滤技术

基于项聚类的协同过滤方法通过聚类的方法把评分相似的项目通过聚类的方式聚在一起，只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。这样就能在尽量少的项目空间上搜索到目标项的最近邻居，有效提高推荐系统的实时响应速度。

K-均值作为最经典的聚类方法在基于项聚类的协同过滤方法中得到了很大的应用，但实验研究表明，由于数据的稀疏性对聚类效果产生了很大的影响，因此K-均值算法初始值选取敏感和容易陷入局部最优的缺点也对推荐的效率产生了影响。

PSO算法是在鸟群、鱼群和人类社会行为规律启发下提出的一种基于群体智能的优化算法技术[2]。在PSO算法中，粒子通过不断调整自己的位置 $X$ 来搜索新解，每个粒子记住自己搜索到的最好解，记为 $P_{id}$，同时整个粒子群经历的最优位置记为 $P_{gd}$，每个粒子都有一个速度 $V$，并通过以下公式更新速度和位置：

$$V_{id}^{'}=\omega V_{id}+\eta_1 rand()\left(P_{id}-X_{id}\right)+\eta_2 rand()\left(P_{gd}-X_{id}\right) \quad (1)$$

$$X_{id}^{'}=X_{id}+V_{id} \quad (2)$$

其中，$V_{id}$ 表示第 $i$ 个粒子在 $d$ 维的速度；$X_{id}$ 表示第 $i$ 个粒子在 $d$ 维的位置；$\omega$ 为惯性权重；$\eta_1,\eta_2$ 为调节参数；$rand()$ 为随机函数；$P_{id},P_{gd}$ 为相对重要的参数。

本文针对推荐系统的缺点和聚类方法存在的缺陷采用PSO算法进行优化。在数据的稀疏性方面采用评分均值填充的方式降低数据的稀疏性。

## 3 基于粒子群优化的项聚类推荐算法

推荐过程一般有5个步骤：(1)形成用户-项目矩阵，这是该算法最根本的数据结构。(2)度量相似性。(3)根据PSO算法产生聚类，计算邻居集合。(4)选择预测公式，产生预测。

**基金项目：**教育部留学回国人员启动基金资助项目(教外司留[2007]1108-10)

**作者简介：**熊忠阳(1964－)，男，教授、博士生导师，主研方向：网格与并行处理，数据挖掘；张凤娟，硕士研究生；张玉芳，副教授

**收稿日期：**2009-04-01 **E-mail：**zoe0528@163.com

(5)形成推荐。本文按照这 5 个步骤介绍粒子群优化的项聚类推荐算法。

(1)形成用户-项目矩阵

用户评分数据可以用一个 $m\times n$ 的矩阵 $\boldsymbol{A}(m, n)$表示，其中，$m$ 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。表示如下：

$$\boldsymbol{A}=\begin{bmatrix} R_{1,1} & \cdots & R_{1,j} & \cdots & R_{1,n} \\ \vdots & & & & \vdots \\ R_{i,1} & \cdots & R_{i,j} & \cdots & R_{i,n} \\ \vdots & & & & \vdots \\ R_{m,1} & \cdots & R_{m,j} & \cdots & R_{m,n} \end{bmatrix}$$

用户可以代表浏览网站的客户或者购买商品的顾客等，项目代表浏览的网页、网站的商品或者电影、新闻等，用户的评分分为显式评分与隐式评分 2 种[3]，显式评分就是通过专门的网页直接请求用户显式输入对某些项目的数值评分，隐式评分则不需要用户直接提供对项目的评分，而是根据用户浏览网页时的行为特征预测用户对该项目信息的评分。不论是何种形式的用户与项目，是显式评分还是隐式评分，经过对数据的初始化处理以后，都可以表示为用户-项目矩阵这种数据结构，以便后续进行推荐。

(2)度量相似性

计算项与项之间的相似性与计算用户之间的相似度相似，主要有基于余弦的相似性、基于相关性的相似性以及调整的余弦相似性等方法。

余弦相似性的公式如下：

$$sim(\boldsymbol{i},\boldsymbol{j})=\cos(\boldsymbol{i},\boldsymbol{j})=\frac{\boldsymbol{i}\cdot\boldsymbol{j}}{\|\boldsymbol{i}\|*\|\boldsymbol{j}\|} \tag{3}$$

由于余弦相似性简单易处理，因此本文选择余弦相似性。为了提高推荐的精度，对矩阵进行简单的处理，引入用户的平均评价值，对未评分的数据做初步处理。原因是在采用 K-均值计算聚类的中心时，如果直接计算，由于矩阵的稀疏性，未评分的数据都作为 0，因此项目的每一维都趋近于 0，聚类中心也朝 0 趋近，大大降低了推荐的精度。为了解决用户评分的极端稀疏性，最简单的办法是将用户对未评分项的评分设为一个固定的缺省值(一般设为评分域的中间值，如在 5 分制中评分设为 3)。本文将未评分项目设为该项目的平均评分，这相当于修正了余弦相似性，降低了计算的复杂度。

(3)计算邻居集合

本文采用基于项目聚类的最近邻查询，分如下 2 步实现：

1)项目聚类

根据上述计算相似性的方法，对项目进行聚类。将数据对象分为多个类，使类内的项目具有较高的相似性，而类间的项目有较大差别。

聚类算法的数学描述如下：

设样本集合为 $X=\{\boldsymbol{X}_i,\ i=1,2,\cdots,n\}$，其中，$\boldsymbol{X}_i$ 为 $d$ 维向量，聚类就是找到一个划分 $C=\{C_1,C_2,\cdots,C_s\}$，满足：

$$X=\bigcup_{i=1}^{s}C_i \tag{4}$$

并且使总的类内离散度和

$$J_c=\sum_{k=1}^{s}\sum_{x_i\in c_k}d(\boldsymbol{X}_i,Z_k) \tag{5}$$

达到最小[4]，其中，$Z_k$ 为第 $k$ 个聚类的中心；$d(\boldsymbol{X}_i,Z_k)$ 为样本与对应聚类中心的相似度；$J_c$ 为各类样本到对应聚类中心距离的总和。由于应用于推荐系统，因此这里 $d(\boldsymbol{X}_i,Z_k)$ 为相似度，即 $d(\boldsymbol{X}_i,Z_k)=sim(\boldsymbol{X}_i,Z_k)$。

本文采用粒子群优化的聚类算法对样本进行聚类，将类内离散度和 $J_c$ 作为适应度函数 $f=\sum_{k=1}^{s}\sum_{x_i\in c_k}d(\boldsymbol{X}_i,Z_k)$，每个粒子的位置由 $s$ 个聚类中心组成，样本的维数为 $m$，因此，每个粒子为 $s\times m$ 的向量。采用粒子群优化算法进行聚类改进，具体的算法实现描述如下：

**输入** 用户评分矩阵 $\boldsymbol{A}(m, n)$，聚类个数 $s$，粒子数目 $k$

**输出** $s$ 个聚类及其中心

①将评分矩阵中的 $n$ 个项目记为集合 $I=\{I_1,I_2,\cdots,I_n\}$。

②种群初始化，随机生成一个 $s\times m$ 的粒子，将 $n$ 个项目分到粒子中，其中与某个聚类中心最相似的就分到这个聚类中，重复 $k$ 次，产生 $k$ 个粒子。每个粒子的当前位置为 $P_{id}$，$P_{gd}$ 为所有粒子的最优位置。

③按照式(1)、式(2)对粒子的位置和速度进行更新，产生新一代的粒子。

④按式(5)计算每个粒子的适应度，如果好于该粒子的当前最优解 $P_{id}$，则更新该粒子的当前最优解。如果所有粒子全局最优解好于当前的最优解 $P_{gd}$，则更新全局最优解。

⑤重复③，直到达到最大迭代次数或足够好的位置。

⑥输出全局最优位置 $P_{gd}$，即为最优的聚类中心。

用上述方法可以得到聚类中心的评分矩阵。用这个矩阵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几个聚类作为搜索空间，在此空间搜索最近邻居。由于项目聚类比较费时，因此可以离线进行。

2)最近邻搜索

经过项目聚类后，在目标项相似性最高的几个聚类中就能找到目标项目的大部分邻居，而不用在整个空间内搜索。与目标项目相似性最高的几个聚类空间相对于整个项目空间小得多，因此，可以大大提高在线的搜索速度，使推荐系统的实时性得到保证。

最近邻搜索描述如下：

①计算目标项与每个聚类中心的相似性。

②选择小于相似度阈值的聚类中心所在的聚类进行搜索，计算聚类内项目与目标项目的相似性。

③找出与目标项目最相近的前 $N$ 个邻居作为目标项目的最近邻居。

在进行最近邻搜索时，可自动对相似度阈值进行调整。当推荐系统探测到推荐精度降低时(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内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较长时，可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。

(4)产生预测

计算出目标项目的最近邻居后，计算用户对项目的预测评分。设目标项目 $p$ 的最近邻居集合为 $NN_p=\{NN_1,NN_2,\cdots,NN_K\}$，用户 $u$ 对项目 $p$ 的预测方法如下[1]：

$$P_{u,p}=\overline{R_p}+\frac{\sum_{n\in NN_p}sim(p,n)\times(R_{u,n}-\overline{R_n})}{\sum_{n\in NN_p}(|sim(p,n)|)}$$

其中，$sim(p,n)$ 表示目标项目 $p$ 与最近邻居 $n$ 的相似性；$R_{u,n}$ 代表用户 $u$ 对项目 $n$ 的评分；$\overline{R_p}$ 和 $\overline{R_n}$ 分别代表项目 $p$ 和项目 $n$ 的平均评分。

(5)形成推荐

用上述方法计算当前用户对所有未评分项目的预测评分，然后选择评分最高的前 $N$ 个项目推荐给用户。

## 4 实验结果及分析

### 4.1 实验数据集和评价标准

实验采用 MovieLens 站点提供的数据集(http://www.movielens.org/)。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(1 分~5 分)，并提供相应的电影推荐列表。该站点已拥有 43 000 个用户和 3 500 部电影。随机选择 100 000 个评分，组成 943×1 682 的评分矩阵。其中每个用户至少对 20 部电影进行了评分。

本文以常用的推荐质量度量方法平均绝对误差 *MAE* 作为评价标准。平均绝对误差通过计算预测的用户评分和实际的用户评分之间的偏差来度量预测的准确性，*MAE* 越小，推荐质量越高。计算公式如下[5]:

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

其中，预测的用户评分集为 $\{p_1,p_2,\cdots,p_N\}$；实际的用户评分集合为 $\{q_1,q_2,\cdots,q_N\}$。

### 4.2 实验结果及分析

实验中指定项目的聚类数目为 40，因为在聚类中一般聚类数目为项目数开平方取整，在最近邻搜索过程中，取与目标项目最近的前 6 个项目进行聚类，目标用户的最近邻居个数为 10~50，间隔为 10。为比较实验效果，将本文算法与推荐系统中引用比较多的文献[6]的实验数据进行比较，另一个实验结果为文献[5]提出的在全部邻居集上搜索的 Item-based 推荐算法，实验结果如图 1 所示。可以看出，本文的算法具有较小的 *MAE*，即具有较高的推荐质量。由于对数据进行了初始化处理，即用项目的平均评分填充未评分项目，因此预测结果比较平均，*MAE* 都在 0.827 左右。因此，本文算法较好地提高了推荐质量。

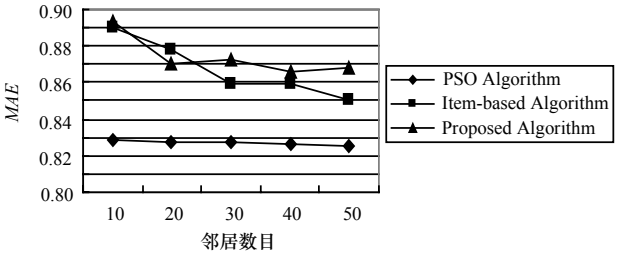


**图 1　3 种算法 *MAE* 比较结果**

## 5 结束语

针对电子商务推荐系统中随商品数量与用户数量的增加而暴露出的系统实时性与推荐质量之间的矛盾，本文采用基于项目聚类的协同过滤算法，将相似性与目标项目较高的项目作为搜索空间，提高了推荐响应速度，并利用粒子群优化算法这种全局寻优的算法对聚类算法进行了改进，实验表明本文算法具有较高的推荐质量。下一步将对 PSO 算法做进一步的研究。

### 参考文献

[1] Breese J, Hecherman D, Kadie C. Empirical Analysis of Predictive Algorithms for Collaborative Filtering[C]//Proceedings of UAI'98. [S. l.]: ACM Press, 1998: 43-52.

[2] Kennedy J, Eberhard R C, Shi Yuhui. Swarm Intelligence[M]. San Francisco, USA: Moraga Kaufman Publisher, 2001: 1942-1948.

[3] Herlocker J. Clustering Items for Collaborative Filtering[C]// Proceedings of the ACM SIGIR Workshop on Recommender Systems. [S. l.]: ACM Press, 2002.

[4] 刘向东，沙秋夫，刘勇奎，等. 基于粒子群优化算法的聚类分析[J]. 计算机工程, 2006, 32(6): 201-202.

[5] Sarwar B, Karypis G, Konstanz J, et al. Item-based Collaborative Filtering Recommendation Algorithms[C]//Proceedings of the 10th International World Wide Web Conference. New York, USA: ACM Press, 2001.

[6] 邓爱林，左子叶，朱扬勇. 基于聚类的协同过滤推荐算法[J]. 小型微型计算机系统, 2004, 25(9): 1665-1670.

编辑　张　帆

(上接第 177 页)

本文提出了相对决策条件的概念，通过构建决策类之间的分明矩阵得到决策类之间的相对决策条件，再对相对决策条件进行合并得出所有决策类的决策条件。相对决策条件的计算是通过构建规模较小的分明矩阵来实现的，因此，能够有效地减少运算规模和节省存储空间。另外，多个决策类之间的相对决策条件的计算可以并运计算，这将进一步加快决策表决策规则的提取速度。本文提出的规则提取方法未先对属性约简，它能找出所有规则的约简，并最终得到所有可能的最小化决策规则集，这对实际应用中不完备信息系统的决策具有重要的意义。

### 参考文献

[1] Pawlak Z. Rough Sets[J]. International Journal of Computer and Information Science, 1982, 11(5): 341-356.

[2] 刘　清. Rough 集及 Rough 推理[M]. 北京: 科学出版社, 2001.

[3] Skowron A, Rauszer C. The Discernibility Functions Matrix and Functions in Information Systems[M]. [S. l.]: Kluwer Academic Publisher, 1992.

[4] Wang Jue, Wang Ju. Reduction Algorithms Based on Discerniblility Matrix: The Ordered Attributed Method[J]. Journal of Computer Science and Technology, 2001, 11(6): 489-504.

[5] 黄文涛. 基于粗糙集理论的故障诊断决策规则提取方法[J]. 中国电机工程学报, 2003, 23(11): 151-155.

编辑　金胡考