基于共同评分和相似性权重的协同过滤推荐算法

汪 静[1] 印 鉴[1] 郑利荣[2] 黄创光[1]
(中山大学信息科学与技术学院 广州 510275)[1] (广东医学情报研究所 广州 510180)[2]

摘 要 协同过滤推荐算法是在电子商务推荐系统中应用最成功的推荐技术之一。提出了一种基于共同评分和相似性权重的协同过滤推荐算法。该算法选择用户的共同评分数据计算用户的相似性,选择项目被用户共同评分的数据计算项目的相似性,再分别计算基于用户以及项目算法的预测评分,然后通过相似性权重结合两者得到最终的预测结果,最后再根据预测结果产生推荐。实际数据的实验结果表明,提出的算法显著提高了预测准确度,从而提高了推荐质量。

关键词 电子商务,推荐系统,协同过滤,共同评分,相似性权重
中图法分类号 TP393 **文献标识码** A

Collaborative Filtering Recommendation Algorithm Based on Co-ratings and Similarity Weight

WANG Jing[1] YIN Jian[1] ZHENG Li-rong[2] HUANG Chuang-guang[1]
(Department of Information Science and Technology, Sun-Yat Sen University, Guangzhou 510275, China)[1]
(Guangdong Institute of Medical Information, Guangzhou 510180, China)[2]

Abstract Collaborative filtering recommendation algorithm is one of the most successful technologies in the e-commerce recommendation system. This paper presented a collaborative filtering algorithm based on co-ratings and similarity weight. First, the co-ratings were selected to compute the similarity between users or items. Most importantly, the algorithm acquiring the last prediction result was acquired by combining prior predicting rating with similarity weight, from which recommendation was produced. The experimental results in real data show this algorithm can consistently achieve better prediction accuracy, thereby brings better recommendation quality.

Keywords E-commerce, Recommendation system, Collaborative filtering, Co-rating, Similarity weight

1 引言

为了解决"信息过载"的问题,很多大型电子商务网站,如Amazon,Ebay,阿里巴巴,当当网等都使用了各种形式的推荐系统。电子商务推荐系统可以基于销售排行、用户的兴趣爱好和用户对所浏览商品的评分等来进行推荐。用户对所浏览商品的评分直接反映了用户对商品的喜好程度,据此产生的推荐常常能达到更准确的推荐效果。协同过滤推荐算法正是基于用户对项目的评分数据来产生推荐。至今为止,协同过滤推荐算法是在电子商务推荐系统中应用最成功的推荐技术之一[1-8]。

现有的协同过滤推荐算法主要分为两类:(1)基于近邻(基于内存)的算法,该算法根据相似邻居来产生推荐,相似性邻居可以是用户或项目;(2)基于模型的算法,该算法通过建立评分模型来产生推荐。已有的研究指出,基于近邻的算法能获得更好的推荐准确率,但是无法解决由于数据量增大而带来的可伸缩性问题[9];基于模型的算法有更好的伸缩性,但是由于模型不能表现用户兴趣爱好的多样性,因此在推荐质量方面不如基于近邻的算法[10]。

基于近邻(基于内存)的算法,又可以分为基于用户(UB)[1,2,11,12]和基于项目(IB)[6,9,13-15]的算法。基于用户的算法根据用户的相似性来产生推荐,基于项目的算法根据项目的相似性来产生推荐。基于用户或项目的算法都只是考虑了用户或项目单方面的影响,忽略了用户与项目之间的联系。另一方面,基于用户或项目的算法从所有的评分数据来考虑用户或项目的影响,忽略了在部分数据上用户或项目也可能相似的情况。

本文围绕这些问题展开研究,提出了一种基于共同评分和相似性权重的协同过滤推荐算法(Collaborative Filtering Recommendation Algorithm Based on Co-Ratings and Similarity Weight,简称CRSW)。本文利用用户的共同评分数据来计算用户的相似性,用户对项目的共同评分数据来计算项目

到稿日期:2009-03-26 返修日期:2009-06-10 本文受国家自然科学基金(60573097,60773198,60703111),广东省自然科学基金(05200302,06104916),广州市科技计划项目(2007Z3-D3071),高等学校博士学科点专项科研基金(20050558017),新世纪优秀人才支持计划(NCET-06-0727)资助。

汪 静(1980—),女,博士生,讲师,主要研究领域为个性化推荐等,E-mail:jingyun_wj@163.com;印 鉴(1968—),男,博士,教授,博士生导师,主要研究领域为数据挖掘、电子商务等;郑利荣(1971—),女,硕士生,副研究员,主要研究领域为信息检索、数据挖掘等;黄创光(1978—),男,博士生,主要研究领域为客户行为研究等。

的相似性，再分别计算基于用户以及项目算法的预测评分，然后通过相似性权重结合两者得到最终的预测结果，最后再根据预测结果产生推荐。实验结果表明，本文提出的算法大大提高了预测准确率，可以获得更好的推荐质量。

本文第 2 节对协同过滤推荐算法的相关研究工作做一简要介绍；第 3 节阐述基于共同评分和相似性权重的协同过滤推荐算法；第 4 节是对提出的算法进行实验验证与结果分析；最后是结论与展望。

2 相关工作

基于用户的算法通过分析不同用户的评分，为当前用户寻找若干个最相似的邻居；对于当前用户尚未评分的项目，根据邻居用户对该项目的评分对其作出预测。这种算法主要关注用户与用户兴趣之间的关联性，因而被称作基于用户的协同过滤推荐算法。基于用户的算法最早是由 Goldberg 等在构建邮件过滤系统 Tapestry 时提出的[7]，后来用于 Group Lens 向用户推荐新闻[2]。Herlocker 等最早提出了用户相似性的调整参数和邻居用户的选取阈值，并通过实验证明引入这些参数后提高了推荐准确度[11]。Hee 等提出利用用户相匹配的评分数据来预测用户对未评分项目的评分，实验结果表明改进后的算法获得了更高的推荐质量[12]。

基于项目的算法最早由 Sarwar 等提出[9]。该算法为当前用户尚未评分的项目寻找若干个最相似的项目，然后利用当前用户对相似性项目的评分来预测当前用户对尚未评分的项目的评分。Amazon. com 网站利用该算法为用户推荐商品，获得了更好的实时推荐效果[6]。Deshpande 等通过实验结果说明该算法的推荐质量高于基于用户的算法[13]。陈健等提出了基于影响集的协同过滤算法，此算法通过寻找项目的影响集来增加用户对项目的评分数据，提高了推荐质量[15]。

最近几年对协同过滤推荐算法的研究集中在如何结合基于用户和基于项目的算法。Xue 等提出基于聚类的协同过滤推荐算法，利用聚类技术进行空缺值的填补和用户邻居以及项目邻居的选取[10]。Wang 等通过建立概率模型，来预测空缺的用户评分数据，并提出了融合基于用户和基于项目算法的预测值的计算方法[16]。Ma 等提出了一种基于用户和基于项目的加权算法来填充空缺值，然后预测协同过滤推荐算法，并通过实验证明其推荐质量优于其他算法[17]。这些算法考虑了用户和项目之间的联系，但是没有考虑在部分数据上用户或项目也可能相似的情况。而且这些算法都是以固定权重结合基于用户和基于项目的算法，在对数据集事先没有充分了解的前提下难以给出恰当的固定权重，而且固定权重也不能准确表示基于用户和基于项目的算法对预测结果的影响程度。本文提出的基于共同评分和相似性权重的协同过滤推荐算法既考虑了在部分数据上用户或项目相似的情况，也解决了算法对固定权重敏感的问题。

3 基于共同评分和相似性权重的协同过滤推荐算法

3.1 问题描述

在基于协同过滤推荐算法的推荐系统中，用户评分数据库中包含 m 个用户的集合 $U=\{u_1,u_2,\cdots,u_m\}$ 和 n 个项目的集合 $I=\{i_1,i_2,\cdots,i_n\}$。用户对项目的评分数据可以采用一个 $m\times n$ 阶的用户-项目评分矩阵 $R(m,n)$ 来表示，如表 1 所列。

表 1 用户-项目评分矩阵 $R(m,n)$

	i_1	…	i_t	…	i_n
u_1	$R_{1,1}$	…	$R_{1,t}$	…	$R_{1,n}$
…					
u_a	$R_{a,1}$	…	$R_{a,t}=?$	…	$R_{a,n}$
…					
u_m	$R_{m,1}$	…	$R_{m,t}$	…	$R_{m,n}$

其中，评分表示用户对项目感兴趣的程度，评分的级别越高，说明用户越感兴趣。

3.2 基于共同评分数据的相似性

相似性计算是协同过滤推荐算法中最关键的步骤。文献[11,12]通过实验表明 Pearson 相关系数能更好地表示用户或项目的相似程度，本文也采用了 Pearson 相关系数。

3.2.1 用户的相似性

用户的相似性表示用户兴趣爱好的相似程度。用户的共同评分数据可以表示他们在某一方面兴趣爱好的相似程度，计算如下：

$$\mathrm{Sim}(a,u)=\frac{\sum_{i\in I(a)\cap I(u)}(R_{a,i}-\overline{R}'_a)(R_{u,i}-\overline{R}'_u)}{\sqrt{\sum_{i\in I(a)\cap I(u)}(R_{a,i}-\overline{R}'_a)^2}\sqrt{\sum_{i\in I(a)\cap I(u)}(R_{u,i}-\overline{R}'_u)^2}} \tag{1}$$

其中，$\mathrm{Sim}(a,u)$ 表示用户 a 和 u 的相似性，i 表示用户 a 和 u 共同评分的项目，$I(a)$ 表示用户 a 评分的项目集合，$I(u)$ 表示用户 u 评分的项目集合，i 属于 $I(a)$ 和 $I(u)$ 的交集，$R_{a,i}$ 表示用户 a 对项目 i 的评分，$R_{u,i}$ 表示用户 u 对项目 i 的评分。$\overline{R}'_a$ 和 $\overline{R}'_u$ 分别表示在共同评分数据中用户 a 和 u 的评分均值，计算如下：

$$\overline{R}'_a=\frac{\sum_{i\in I(a)\cap I(u)}R_{a,i}}{k} \tag{2}$$

$$\overline{R}'_u=\frac{\sum_{i\in I(a)\cap I(u)}R_{u,i}}{k} \tag{3}$$

其中，$k=|I(a)\cap I(u)|$，是 $I(a)$ 和 $I(u)$ 的交集的元素个数，表示用户 a 和 u 共同评分的项目个数。

在选择用户的共同评分数据来计算用户的相似性时，可能会出现这样的情况：两个用户共同评分的项目个数很少，但是由于他们的评分非常接近，因此计算的相似性很高。实际上这种情况可能会高估用户的相似性，因为如果两个用户同时感兴趣的项目个数很少，那么他们的兴趣爱好很可能不相似。所以，在选择共同评分数据的前提下，还要考虑共同评分的项目个数，这样才能更准确地表示用户兴趣爱好的相似程度。因而本文利用用户共同评分的项目个数来调整用户相似性，表示如下：

$$\mathrm{Sim}'(a,u)=\frac{\mathrm{Min}(k,\gamma)}{\gamma}\cdot\mathrm{Sim}(a,u) \tag{4}$$

其中，k 表示用户 a 和 u 共同评分的项目个数，γ 表示为调整用户的相似性而预先设定的参数。$\mathrm{Min}(k,\gamma)$ 表示取 k 和 γ 中数值较小的一个，如果用户 a 和 u 共同评分的项目个数大于或等于 γ，则不需要调整用户的相似性；反之，则以 k 和 γ 的比值来调整用户的相似性。

3.2.2 项目的相似性

项目的相似性表示用户对项目同时感兴趣的程度。如果

不同的用户对两个项目的共同评分趋向一致，说明他们对这两个项目的感兴趣程度也趋向一致。选择不同用户对项目的共同评分数据来计算项目的相似性，表示如下：

$$\mathrm{Sim}(i,j)=\frac{\sum_{u\in U(i)\cap U(j)}(R_{u,i}-\overline{R}'_i)(R_{u,j}-\overline{R}'_j)}{\sqrt{\sum_{u\in U(i)\cap U(j)}(R_{u,i}-\overline{R}'_i)}\sqrt{\sum_{u\in U(i)\cap U(j)}(R_{u,j}-\overline{R}'_j)^2}} \tag{5}$$

其中，$\mathrm{Sim}(i,j)$表示项目 i 和 j 的相似性。u 表示对项目 i 和 j 共同评分的用户，$U(i)$表示对项目 i 评分的用户集合，$U(j)$表示对项目 j 评分的用户集合，u 属于 $U(i)$和 $U(j)$的交集。$R_{u,i}$表示用户 u 对项目 i 的评分，$R_{u,j}$表示用户 u 对项目 j 的评分。$\overline{R}'_i$和$\overline{R}'_j$分别表示基于用户对项目 i 和 j 共同评分数据的项目 i 和 j 的评分均值，分别表示如下：

$$\overline{R}'_i=\frac{\sum_{u\in U(i)\cap U(j)}R_{u,i}}{l} \tag{6}$$

$$\overline{R}'_j=\frac{\sum_{u\in U(i)\cap U(j)}R_{u,j}}{l} \tag{7}$$

其中，$l=|U(i)\cap U(j)|$，是 $U(i)$和 $U(j)$的交集的元素个数，表示对项目 i 和 j 共同评分的用户个数。

两个项目被用户共同评分的情况越少，说明他们同时被用户感兴趣的情况越少，那么他们的相似程度应越低。因而本文利用共同评分的用户个数来调整项目的相似性，表示如下：

$$\mathrm{Sim}'(i,j)=\frac{\mathrm{Min}(l,\delta)}{\delta}\cdot\mathrm{Sim}(i,j) \tag{8}$$

其中，l 表示对项目 i 和 j 共同评分的用户个数，δ 表示为调整项目的相似性而预先设定的参数。$\mathrm{Min}(l,\delta)$表示取值为 l 和 δ 中数值较小的一个，如果对项目 i 和 j 共同评分的用户个数大于或等于 δ，则不需要调整项目的相似性；反之，则以 l 和 δ 的比值来调整项目的相似性。

3.3 相似性邻居的选取

经过调整以后的用户以及项目的相似性可能会较小，如果把这样的用户或项目作为相似性邻居来预测评分，可能会降低预测的准确率，从而影响最终的推荐质量。所以本文引入阈值 η,θ 来分别限定用户以及项目相似性邻居的选取。

3.3.1 用户的相似性邻居的选取

$$S(u)=\{a\mid\mathrm{Sim}'(a,u)>\eta,a\neq u\} \tag{9}$$

其中，$S(u)$表示用户 u 的相似性邻居用户集合，η 表示相似性邻居用户的选取阈值，η 的取值决定了相似性邻居用户集合的个数。只有当其他用户与用户 u 的相似性大于 η 时，才能将其选取为用户 u 的相似性邻居。

3.3.2 项目的相似性邻居的选取

$$S(i)=\{j\mid\mathrm{Sim}'(i,j)>\theta,j\neq i\} \tag{10}$$

其中，$S(i)$表示项目 i 的相似性邻居项目集合，θ 表示相似性邻居项目的选取阈值，θ 的取值决定了相似性邻居项目集合的个数。只有当其他项目与项目 i 的相似性大于 θ 时，才能将其选取为项目 i 的相似性邻居。

3.4 预测评分

3.4.1 基于用户的算法的预测评分

选取当前用户的相似邻居用户以后，就可以根据邻居用户对当前项目的评分来预测当前用户对该项目的评分。基于用户的算法的预测评分可以表示如下：

$$P_{user}(R_{u,i})=\overline{R}'_u+\frac{\sum_{a\in S(u)}\mathrm{Sim}'(a,u)\cdot(R_{a,i}-\overline{R}'_a)}{\sum_{a\in S(u)}\mathrm{Sim}'(a,u)} \tag{11}$$

其中，$S(u)$表示通过阈值 η 选取的用户 u 的相似性邻居用户集合，$\mathrm{Sim}'(a,u)$表示经过参数 γ 调整后的用户 a 和 u 的相似性，$R_{a,i}$表示用户 a 对项目 i 的评分，$\overline{R}'_a$和$\overline{R}'_u$分别表示基于共同评分数据的用户 a 和 u 各自评分的均值，用式(2)、式(3)计算。

3.4.2 基于项目的算法的预测评分

选取当前项目的相似邻居项目后，就可以根据邻居项目获得的评分来预测当前用户对当前项目的评分。基于项目的算法的预测评分表示如下：

$$P_{item}(R_{u,i})=\overline{R}'_i+\frac{\sum_{j\in S(i)}\mathrm{Sim}'(i,j)\cdot(R_{u,j}-\overline{R}'_j)}{\sum_{j\in S(i)}\mathrm{Sim}'(i,j)} \tag{12}$$

其中，$S(i)$表示通过阈值 θ 选取的项目 i 的相似性邻居项目集合，$\mathrm{Sim}'(i,j)$表示经过参数 δ 调整后的项目 i 和 j 的相似性，$R_{u,j}$表示用户 u 对邻居项目 j 的评分，$\overline{R}'_i$和$\overline{R}'_j$分别表示基于用户对项目 i 和 j 共同评分数据的项目 i 和 j 的评分均值，用式(6)、式(7)计算。

3.4.3 对基于用户和基于项目的算法的动态加权

为了结合基于用户和基于项目的算法，本文引入了权重因子 λ_u 和 λ_i，它们都在[0,1]上取值，λ_u 表示基于用户的算法对预测结果的动态影响程度，λ_i 表示基于项目的算法对预测结果的动态影响程度。通过自动选择合适的权重因子，可以动态结合基于用户和基于项目的算法各自的有利因素，使预测结果达到一个良好的平衡，从而使推荐更准确也更稳定。对权重因子 λ_u 和 λ_i 的动态取值考虑如下：

(1)当用户的相似性邻居个数越多，而且他们的相似性越强时，基于用户的算法对预测结果的影响越大，此时 λ_u 的值应越大；反之，λ_i 的值应越大。

(2)由于不同用户的相似性不同，因此预测不同用户的评分时，λ_u 的值应该不同；同理，由于不同项目的相似性也不同，因此预测用户对不同项目评分时，λ_i 的值也应该不同。

(3)由于用户以及项目的相似性体现了动态变化的特征，与上面所述的(1)、(2)相符，因此本文利用用户相似性平方和与项目相似性平方和的比值来动态设定 λ_u 和 λ_i，对相似性平方是为了增强它们的影响程度，使预测结果更倾向于由相似性较强的一方获得，用公式表示如下：

$$\lambda_u=\frac{\sum_{a\in S(u)}\mathrm{Sim}'^2(a,u)}{\sum_{a\in S(u)}\mathrm{Sim}'^2(a,u)+\sum_{j\in S(i)}\mathrm{Sim}'^2(i,j)} \tag{13}$$

$$\lambda_i=\frac{\sum_{j\in S(i)}\mathrm{Sim}'^2(i,j)}{\sum_{a\in S(u)}\mathrm{Sim}'^2(a,u)+\sum_{j\in S(i)}\mathrm{Sim}'^2(i,j)} \tag{14}$$

其中，$\lambda_u+\lambda_i=1$。经过动态加权后计算预测结果的公式表示如下：

$$P(R_{u,i})=\lambda_u\times P_{\mathrm{user}}(R_{u,i})+\lambda_i\times P_{\mathrm{item}}(R_{u,i}) \tag{15}$$

3.5 推荐

计算得到当前用户对未评分项目的预测评分后，就可以选择预测评分最高的若干项，推荐给当前用户，这也是目前的推荐系统中常用的 Top-N 推荐。

4 实验结果及分析

实验平台使用 ThankpadX60 Intel(R) Core(TM)2 CPU

T5500 @ 1.66GHz,1G 内存,操作系统为 Windows XP,算法使用标准 C++编写。

4.1 数据集

本文实验采用的数据集是 MovieLens 站点(http://movielens.umn.edu)提供的数据集。这个数据集由美国 Minnesota 大学的 GroupLens 研究小组创建并维护。我们选取的是 100k 的公开数据集(1997 年 9 月 19 日－1998 年 5 月 22 日的数据集,这是目前在衡量推荐算法质量中最常用的数据集),包括 943 个用户对 1682 部电影的评分记录。其中,每个用户至少对 20 部电影进行了评分。评分的范围是从 1～5,5 表示最喜欢,1 表示最不喜欢,用户通过评分的数值表达了自己的兴趣爱好。实际评分数据的密度为 100000/ 943 * 1682=6.3%,数据集相当稀疏。

4.2 度量

本文实验采用平均绝对偏差 MAE(Mean Absolute Error)作为度量算法好坏的标准。MAE 通过计算预测评分与实际评分之间的偏差来度量预测的准确性,它是协同过滤推荐算法中最常用的一种推荐质量度量方法[18]。MAE 越小,表明预测越准确,推荐质量越高。

$$MAE=\frac{\sum_{i=1}^{N}|R_{u,i}+\hat{R}_{u,i}|}{N} \tag{16}$$

其中,$R_{u,i}$表示用户 u 对项目 i 的实际评分,$\hat{R}_{u,i}$表示根据推荐算法预测的用户 u 对项目 i 的评分。N 表示预测评分的个数。

4.3 实验过程及结果比较

本文实验从数据集中抽取了 500 个用户。为了增加评分数据的密度,本文首先填充这 500 个用户对未评分项目的评分,再把它们分为两组,选择 300 个用户作为训练集(训练集大小依次变化为 300,200,100 个用户),剩下的 200 个用户作为测试集。然后采用只给出测试用户的 5 个实际评分、10 个实际评分和 20 个实际评分(Given5,Given10 和 Given20)的方式,分别预测测试用户对未给出评分项目的评分。最后计算预测评分与实际评分的差值,通过 MAE 来度量预测的准确度。本文实验参数设置为 $\gamma=\delta=40,\eta=\theta=0.55$。

为了表明 CRSW 算法的性能,将本文的实验和最近几年提出的协同过滤推荐算法,包括 EMDP(2007 年)[17],SF(2006 年)[16],SCBPCC(2005 年)[15]进行了比较,结果如图 1 所示。

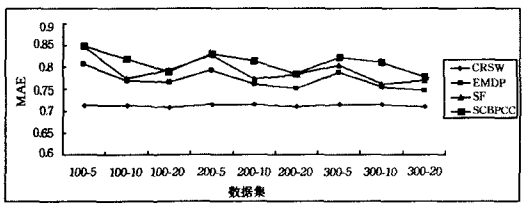

图 1 CRSW 算法与其它协同过滤算法的 MAE 比较

从图 1 的结果可以看出,不论什么情况下,CRSW 算法的 MAE 都明显低于其他算法,这说明该算法预测的评分更接近于用户实际的评分,而且,其 MAE 的变化比较平稳,说明该算法的性能比较稳定,受数据集大小的影响较小。

4.4 参数取值分析

4.4.1 γ 和 δ

γ 和 δ 表示为调整用户以及项目的相似性分别预先设定的参数,这两个参数使得用户以及项目的相似性的计算更加合理,利用它们可以调整那些用户共同评分数据很少但是相似性却很高的情况。图 2 表示 γ 和 δ 对 MAE 的影响,图 3 表示 γ 和 δ 对预测率的影响。预测率指实际预测的个数与需要预测的个数的比率,本文利用它表示 CRSW 算法计算预测评分的有效性。图 4 的曲线表示 γ 和 δ 对数据密度的影响。实验结果表明:(1)随着 γ 和 δ 的增加,MAE 逐渐减小,这说明预测评分越来越接近实际评分。(2)随着 γ 和 δ 的增加,预测率在逐渐减小,这说明能够预测的数据逐渐变少。(3)随着 γ 和 δ 的增加,可以填充的数据逐渐减少,说明数据的密度也逐渐减小。

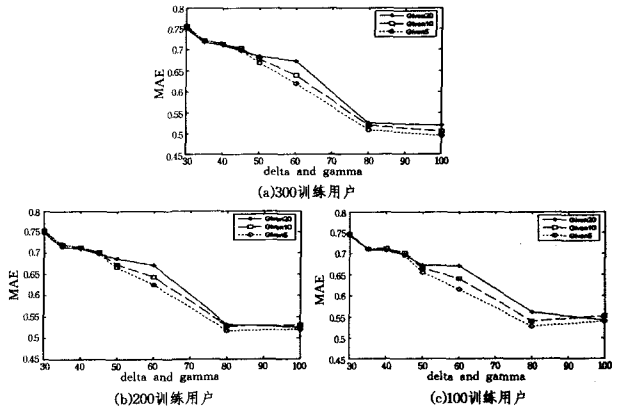

图 2 γ 和 δ 对 MAE 的影响

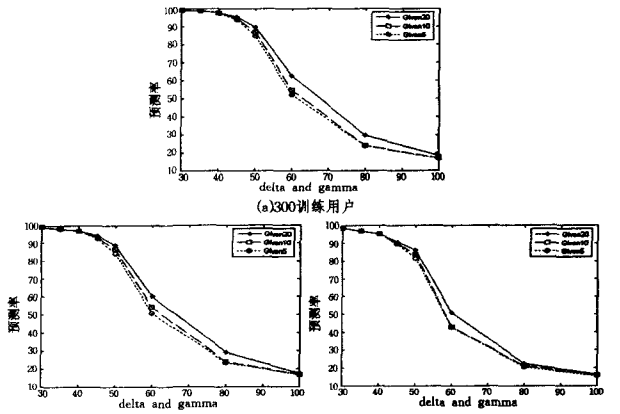

图 3 γ 和 δ 对预测率的影响

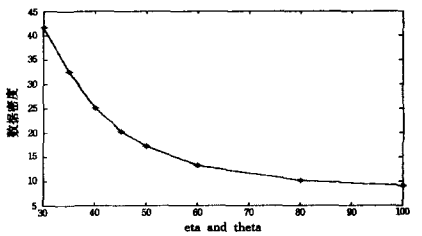

图 4 γ 和 δ 对数据密度的影响

当 γ 和 δ 都等于 40 时,预测率都在 95%以上;而当 γ 和 δ 大于 40 时,预测率降低较快。所以,γ 和 δ 都取 40 可以使该算法从预测的准确率和范围上都取得很好的效果,而此时的数据密度为 25.1%,相对于原始数据的密度已经增加了 3 倍,在一定程度上缓解了由于大量空缺值而影响预测准确率的情况。

4.4.2 η和θ

η和θ表示用户以及项目的相似性邻居选取阈值，这两个参数直接决定了用户以及项目的相似性邻居集合的元素个数。如果对η设置太低，将会增加用户的相似性邻居个数，从而降低基于用户的算法的预测准确率，而且会增加计算的复杂度。同理，如果对θ设置太低，也会影响基于项目的算法的预测准确率，而且由于项目的数量远远大于用户的数量，因此会大大增加计算的复杂度。为了分析η和θ的影响，设置参数$\gamma=\delta=40$，然后让η和θ从0.4变化到0.75，每次增加0.05。图5表示η和θ对MAE的影响。图6表示η和θ对预测率的影响。图7的曲线表示η和θ对数据密度的影响。实验结果表明：(1)随着η和θ的增加，MAE逐渐减小，这说明预测评分越来越准确。(2)随着η和θ的增加，预测率逐渐减小，这说明能够预测的数据逐渐变少。(3)随着η和θ的增加，数据的密度迅速降低，下降的幅度远远大于γ和δ对数据密度的影响，这说明η和θ对空缺数据填充的影响更大。

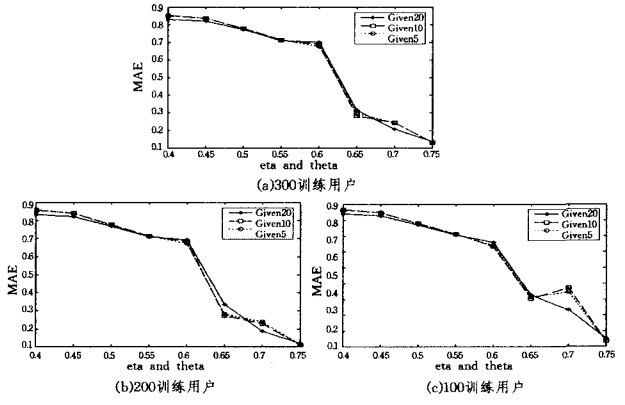

图5 η和θ对*MAE*的影响

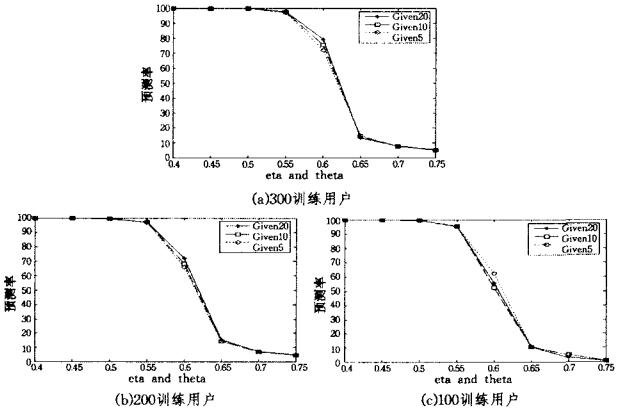

图6 η和θ对预测率的影响

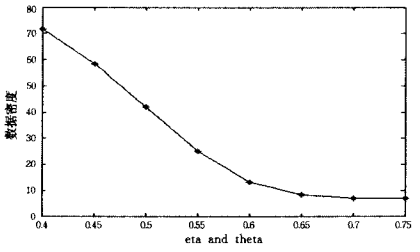

图7 η和θ对数据密度的影响

本文选取了0.55作为参数η和θ的取值，因为此时的预测率均在95%以上，数据的密度为25.1%，而且可以获得较高的预测准确率。

结束语 本文提出了一种基于共同评分和相似性权重的协同过滤推荐算法。本文的主要贡献在于：(1)考虑了在部分数据上用户或项目也可能相似的情况，利用共同评分数据来计算用户以及项目的相似性。(2)考虑了用户和项目之间的联系，利用用户的相似性的平方和与项目的相似性的平方和的比值作为权重，动态结合基于用户和基于项目的协同过滤推荐算法，解决了数据对固定权重敏感的问题，使两种算法的结合达到一个良好的平衡。在MovieLens数据集上的实验结果表明，本文提出的CRSW算法的预测准确率高于其他的协同过滤推荐算法，可以获得更好的推荐质量。

除了用户对项目的评分数据，用户以及项目都有各自的属性，如何联系这些属性来预测当前用户的兴趣爱好，从而提高推荐系统的质量，是今后研究的重要内容。此外，目前对协同过滤推荐算法的研究还有很多不同的思路，如聚类技术、概率模型、本体模型等[19]，如何结合各种不同研究方法各自的优势[20]，也是今后一个很重要的研究方向。

参 考 文 献

[1] Goldberg D, Nichols D, Oki B M, et al. Using collaborative filtering to weave an information Tapestry[J]. Communications of the ACM, 1992, 35(12): 61-70

[2] Resnick P, Iacovou N, Suchak M, et al. GroupLens: An open architecture for collaborative filtering of netnews[C]// Proc. of the ACM CSCW'94 Conf. on Computer Supported Cooperative Work. Chapel Hill: ACM, 1994: 175-186

[3] Shardanand U, Mages P. Social information filtering: Algorithms for automating "Word of Mouth"[C]// Proc. of the ACM CHI'95 Conf. on Human Factors in Computing Systems. New York: ACM Press, 1995: 210-217

[4] Hill M, Stead L, Rosenstein M, et al. Recommending and evaluating choices in a virtual community of use[C]// Proc. of the ACM CHI'95 Conf. on Human Factors in Computing Systems. New York: ACM Press, 1995: 194-201

[5] Claypool M, Gokhale A, Miranda T, et al. Combining content-based and collaborative filters in an online newspaper[C]// ACM SIGIR'99 Workshop on Recommender Systems: Algorithms and Evaluation. Berkeley: ACM, 1999

[6] Linden G, Smith B, York J. Amazon. com recommendations: Item-to-item collaborative filtering[J]. IEEE Internet Computing, 2003, 7(1): 76-80

[7] Holmquist L E, Jacobsson M, Rost M. When media gets wise: Collaborative filtering with mobile media agents[C]// Proc. of the IUI 2006, the 10th Int'l Conf. on Intelligent User Interfaces. Sydney. http://portal. acm. org/, 2006

[8] Park S T, Pennock D M. Applying collaborative filtering techniques to movie search for better ranking and browsing[C]// Proc. of the 13th ACM SIGKDD International Conf. on Knowledge Discovery and Data Mining. New York: ACM, 2007: 550-559

[9] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]// Proc. of 10th Int'l World Wide Web Conf. New York: ACM Press, 2001: 285-295

[10] Xue G R, Lin C X, Yang Q, et al. Scalable collaborative filtering using cluster-based smoothing[C]// Proc. of the 2005 ACM SIGIR Conf. New York: ACM Press, 2005: 114-121

[11] Herlocker L J, Konstan A J, Riedl T J. Empirical analysis of de-

sign choices in neighborhood-based collaborative filtering algorithms[J]. Information Retrieval, 2002, 5(4): 287-310

[12] Lee H C, Lee S J, Chung Y J. A study on the improved collaborative filtering algorithm for recommender system[C]// the 5th International Conf. on Software Engineering Research, Management and Applications. IEEE, 2007: 297-304

[13] Deshpande M, Karypis G. Item-based top-n recommendation algorithms[J]. ACM Transactions on Information Systems, 2004, 22(1): 143-177

[14] 邓爱林,朱扬勇,施伯乐. 基于项目评分预测的协同过滤推荐算法[J]. 软件学报, 2003, 14(9): 1621-1628

[15] 陈健,印鉴. 基于影响集的协同过滤推荐算法[J]. 软件学报, 2007, 18(7): 1685-1694

[16] Wang J, de Vries A P, Reinders J T M. Unifying user-based and item-based collaborative filtering approaches by similarity fusion [C]// Proc. of the 2006 ACM SIGIR Conf. New York: ACM Press, 2006: 501-508

[17] Ma H, King ILyu R M. Effective mssing data prediction for collaborative filtering[C]// Proc. of the 2007 ACM SIGIR Conf. New York: ACM Press, 2007: 39-46

[18] Herlocker L J, Konstan A J, Terveen G L, et al. Evaluating collaborative filtering recommender systems[J]. ACM Transaction on Information Systems, 2004, 22(1): 5-53

[19] Vincent S Z, Boi F. Using hierarchical clustering for learning the ontologies used in recommendation systems[C]// Proc. of the 13th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. New York: ACM Press, 2007: 599-608

[20] Deodhar M, Ghosh J. A framework for simultaneous co-clustering and learning from complex data[C]// Proc. of the 13th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. New York: ACM Press, 2007: 250-259

(上接第 89 页)

200 个节点,Sink 节点位于(0,0),节点初始能量为 2J。

图 9 和图 10 分别显示了网络的总能量消耗和节点存活个数随时间的变化。从图 9 中反映出本文的 QoS 协议在网络总能耗上较 DD 协议有了一定的改善。其中本文基本型 QoS 协议与改进型的区别不是很大,因为在网络节点的密度较大时,基本型建立的带状区域的宽度已经很小,此时改进型根据当前 QoS 约束动态调整带状区域的幅度不是很大,两者之间的传输路径相近,以致于路径的传输能耗改善不大。图 10 中节点存活数量也反映了这种情况,两者的网络存活的节点个数较为相近,但比 DD 协议有了一定的提高。

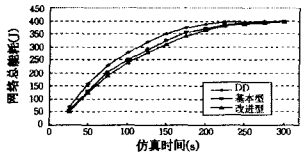

图 9 网络的总能耗随仿真时间的变化

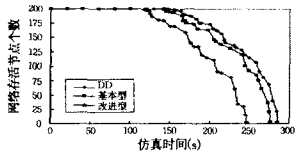

图 10 网络的存活节点数随仿真时间的变化

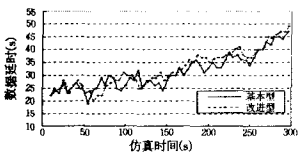

图 11 本文基本型 QoS 协议和改进型的数据延时

图 11 反映了本文基本型 QoS 协议和改进型的数据延时情况。仿真结果显示两者的延时可以分为 3 个阶段的变化:在仿真 0s 到 150s 阶段,两者的延时不是很明显,这一阶段两者几乎都是在 20ms 到 30ms 之间振荡。从图 10 中可以看出,这一阶段还没有发生节点死亡,网络的密度比较大,两者的传输路径基本相似,故延时比较接近。在 150s 到 250s 阶段,已经有相当数量的网络节点失效,网络节点密度发生明显变化,这一阶段基本型在数据延时方面要优于改进型(实线代表基本型,虚线代表改进型)。因为基本型是在源节点初始建立的带状区域内选择最小跳数的下一转发节点,所以传输路径也是整个初始带状区域中的最小跳数路径。而改进型因动态调整带状区域的宽度,可能导致到 Sink 节点为最小跳数的点排除在新建立起来的带状区域内(图 6 就反映了这种情况)。改进型是保证数据延时在允许的范围内,动态调整带状区域宽度寻找一条能耗最小的传输路径,在数据延时上不具有最优。在 250s 到 300s 阶段,数据传输延时加大,可能已经达不到某些任务的延时需求。导致这种情况的原因是,在仿真后期大量的节点已经死亡,建立的带状区域内节点个数已达不到网络 QoS 要求。

结束语 无限传感器网络中,由于节点能量有限,因此传感器网络 QoS 协议设计应在满足 QoS 约束的前提下,尽最大可能降低网络能耗,以延长网络寿命。仿真实验表明,本文基于带状区域路由 QoS 协议在满足 QoS 约束条件下,一定程度地降低了网络能耗,延长了网络生存时间。但协议还存在一些不足,如网络能耗的不均衡导致节点非均匀死亡,引起网络的节点密度不均衡而导致计算 QoS 需求的节点数量误差;另外,当节点大量死亡时,建立的带状宽度非常大或无法建立一个这样的带状区域,导致与普通的路由协议无异。这些问题都是笔者将进一步深入研究的内容。

参 考 文 献

[1] Akyildiz I F, Su W, Sankarasubramaniam Y, et al. Wireless sensor networks: a survey[J]. Computer Networks, 2002, 38(4): 393-422

[2] 李国华,田辉,崔鸿雁,等. 无线传感器网络中一种基于能量策略的路由算法[J]. 电子与信息学报, 2006, 28(1): 168-171

[3] Kay J, Frolik J. Quality of service analysis and control for wireless sensor networks[C]// MOSS. October 2004

[4] Zhou Juejia, Mu Chundi. A Kind of Application Specific QoS Control in Wireless Sensor Networks[C]// Proceedings of the 2006 IEEE International Conference on Information Acquisition. Wei hai, Shandong, China, August 2006: 20-23

[5] Carle G, Biersack E. Survey of error recovery techniques for IP based audio2visual multicast applications[J]. IEEE Network, 1997, 11 (6): 24-36

[6] 李晓维. 无限传感器网络技术[M]. 北京:北京理工大学出版社, 2007

[7] Donoho D L, Grimes C. Hessian eigenmap s : locally linear embedding techniques for high2dimensional data[J]. Science, 2003, 100(10): 5591-5596

[8] Patwari N, Ash J. Locating the nodes : cooperative localization in wireless sensor networks[J]. IEEE Signal Processing Magazine, 2005, 22(4): 54-69

[9] 于斌,孙斌,温暖,等. NS2 与网络模拟[M]. 北京:人民邮电出版社, 2007