文章编号：2096-1472(2019)-06-17-05
DOI:10.19644/j.cnki.issn2096-1472.2019.06.005

基于多种偏置项融合时间信息的协同过滤算法

李明秀[1]，王淑军[2]，贾 如[1]，陈立荣[1]
(1.内蒙古大学计算机学院，内蒙古自治区 呼和浩特 010000；
2.天津大学计算机科学与技术学院，天津 300350)

摘 要：协同过滤算法是实现推荐系统最重要的技术之一。随着时间的推移，用户对物品的偏好会不断地发生变化，物品自身的流行度也会随时间不断地发生变化。目前常用的推荐算法如基于邻域的协同过滤算法itemCF、userCF和隐语义模型算法FunkSVD、BiasSVD、SVD++都没有考虑到时间因素对推荐系统推荐质量的影响。而时间信息是一种非常重要的上下文信息，应该在算法中加以利用。本文使用Sigmoid函数和流行度函数将时间因素融入到了BiasSVD算法中，成功的设计出了一个融合时间信息的新算法Time-BiasSVD。在MovieLens数据集上的验证结果表明：该算法与已有协同过滤算法，以及融合时间信息的算法timeSVD++相比，能更准确地预测用户实际评分，提高推荐系统的推荐质量。

关键词：偏置项；协同过滤；时间因素；Sigmoid函数；流行度函数
中图分类号：TP399 **文献标识码**：A

Collaborative Filtering Algorithm Based on Integrating Time Factors into Multiple Biases

LI Mingxiu[1],WANG Shujun[2],JIA Ru[1],CHEN Lirong[1]
(1.*College of Computer Science & Technology, Inner Mongolia University*,*Huhhot* 010000,*Chna*;
2.*College of Computer Science & Technology*,*Tianjin University*,*Tianjin* 300350,*China*)

Abstract: Collaborative filtering algorithm is one of the most important techniques to implement recommendation system.As time goes by,users' preferences for items will change constantly and the popularity of items will also change over time.At present,the commonly used recommendation algorithms,such as neighborhood-based collaborative filtering algorithm itemCF,userCF and implicit semantic model algorithm FunkSVD,BiasSVD,SVD ++,do not take into account the impact of time factors on the recommendation quality of the recommendation system.In this paper,the Sigmoid function and the popularity function are used to design a new algorithm by integrating the time factor into the BiasSVD algorithm.The verification results on the MovieLens dataset show that the proposed algorithm can accurately predict the user's actual score and improve the recommendation quality of the recommendation system compared with the existing collaborative filtering algorithms and the timeSVD ++ algorithm.

Keywords:bias;time factor;collaborative filtering;Sigmoid function;popularity function

1 引言(Introduction)

随着大数据时代的到来，网络空间中蕴含的信息量几何式增长。在这种背景下，用户如何快速的获得自己需要的信息就变成了一个非常严峻的问题。推荐系统可以帮助用户发现他们可能喜欢的物品，这有效地解决了信息过载的问题。

协同过滤(Collaborative Filtering,简称CF)[1,2]是目前推荐系统中应用最广泛、最成功的技术。目前在工业界中得到最广泛应用的协同过滤算法是基于邻域的协同过滤算法。基于邻域的协同过滤算法包括两种，基于用户相似度[3,4]和基于物品相似度[5]。隐语义模型(Latent Factor Model,简称LFM)[6]使用用户的历史评分数据，挖掘出数据中隐含的特征。时间信息是一种非常重要的上下文信息，模型能反映最新的数据表

基金项目：国家自然科学基金“在线评论偏差对商家信誉和平台收益的影响及纠偏措施研究”，批准号：71862027.

达的内容与特性并且能捕捉数据本质的长期趋势是推荐算法要解决的重要问题。

本文在隐语义模型的基础上提出了一种融合时间信息的新思路，并设计出了融合时间信息的协同过滤算法Time-BiasSVD。其核心思想是：通过构建一条权重变化曲线将用户兴趣按照评分产生的时间赋予不同的权重，使得每条记录对用户的影响不同。距离当前推荐时间越近的记录权重越大，对用户的影响越明显。而物品本身也存在不同时间段内流行度不同的问题，因此物品也是一个和时间相关的量。将Sigmoid函数与物品的流行度融入到当前流行的BiasSVD算法中就实现了本文的Time-BiasSVD算法。实验结果表明，该算法比已有隐语义模型算法能更有效地预测用户的评分。同时相比较于koren等人提出的timeSVD++算法，本文提出的使用Sigmoid函数和流行度函数将时间因素分别融入到用户和物品偏置项的新思路使得推荐系统的推荐精度也有了明显的提高。

2 相关工作(Related research)

2.1 隐语义模型

奇异值分解(Singular Value Decomposition,简称SVD)SVD可以将一个庞大的原始矩阵拆分成为三个小的矩阵[7]。在损失原始矩阵小部分准确度的同时大大降低存储空间的使用。但是SVD也面临稀疏矩阵和时间复杂度过高这两个非常严重的问题[8,9]。

Simon Funk后来提出的FunkSVD又被称为隐语义模型。该算法既减少了对存储空间的利用，又避免了传统SVD矩阵分解面临的时间复杂度过高和稀疏矩阵的问题。

加入偏置项的隐语义模型BiasSVD算法在FunkSVD算法的基础上增加了两个偏置项参数，分别用来代表用户在评分过程中由于用户自身性格等因素产生的误差，以及项目本身属性导致用户实际评分产生的误差。BiasSVD算法将FunkSVD算法中的两个矩阵P和Q引入，并增加基础平均分和两个偏置项参数，这五个参数就组成了BiasSVD算法的预测评分函数。在许多场合的实际应用中发现，BiasSVD算法的推荐准确度高于FunkSVD算法。

2.2 基于时序信息的推荐算法

Koren等人提出的timeSVD++算法[10]将物品偏差进行了分割，给每个分割段使用数值不同的物品偏差，将b_i变成了$b_i(t)$。

与此同时timeSVD++算法也给出了一个使用一次函数给用户偏置项融入时间偏移量的方法，把用户偏置项也变成和时间相关的参数。将用户偏置项b_u转变成了$b_u(t)$。公式为：

$$dev_u(t) = sign(t - t_u) * |t - t_u|^{\beta} \quad (1)$$

$$b_u(t) = b_u + {}^{a}{}_{u} * dev_u(t) \quad (2)$$

其中，t代表数据集中每条记录的评分时间，t_u代表数据集中所有记录评分时间的平均值。a_u和β是两个参数，通过算法训练学习获得。

通过以上方式，timeSVD++较好的实现了在SVD++算法的基础上融合时间信息。

Cigdem BAKIR在SVD算法的基础上融合了时间动态信息[11]。在该算法中考虑了评分的“年龄”，即评分的时间到现在推荐时间的时间差。降低年老评分的权重，增加年轻评分的权重。关键公式如下：

$$Age_w = mAge_r + n \quad (3)$$

在这个公式中Age_r代表评分的年龄，Age_w代表加了权重的评分的年龄，m、n是参数。

$$r_{aged} = \frac{r_{org}}{Age_w} \quad (4)$$

其中，r_{org}代表原始评分，r_{aged}代表权重评分。

孙光福等人提出的SequentialMF算法[12]使用用户的评分数据和评分时间信息来构建用户和产品的消费网络图，并根据图计算影响力最大的近邻集，再把近邻集用到概率矩阵分解模型中然后分别计算出用户和项目的特征向量，根据该特征向量预测重构评分矩阵，再对用户进行推荐。

与上述相关工作的不同之处，本研究基于不同偏置项各自的特点设计出了使用多种偏置项融合时间信息的协同过滤算法Time-BiasSVD，主要贡献如下：一是本文创新的使用了变型的Sigmoid函数，并利用它模拟用户兴趣偏置项随时间的变化趋势；二是本文在项目偏置项中找出了电影流行度和平均评分的关系，进而将物品偏置项中的流行度分离出来，给物品偏置项也融入了时间信息。

3 基于多种偏置项融合时间信息的协同过滤算法(Collaborative filtering algorithm based on integrating the time factor into multiple biases)

3.1 基本定义

定义1(用户-物品评分矩阵)R={ru,i}，R表示用户对物品的评分矩阵。ru,i表示用户u对物品i的评分，为1—5的整数。

定义2(用户-隐含特征矩阵)P={pu,k}，P表示用户与隐含特征之间权重的矩阵。pu,k表示用户u对第k种特征的喜欢程度。取值范围为0—1的浮点数。

定义3(隐含特征-物品矩阵)Q={qi,k}，Q是表示物品在其所属特征中占重要程度的矩阵。qi,k表示物品i在特征k中所占的重要度。取值为0—1的浮点数。

定义4(用户偏置项)b_u代表预测函数中用户的评分中用户自身性格等因素产生的误差。

定义5(项目偏置项)b_i代表物品本身属性导致的评分误差。

定义6(用户兴趣随时间跨度增大变化函数)$f_u(t)$代表一个物品对用户的刺激随着时间跨度的增大而呈现出的S型下降趋势。数据集中每个评分记录都会有一个时间戳，t代表用户历史评分记录中的时间戳和当前为用户推荐物品的时间戳之间的差值。

定义7(物品在所属时间片内的流行度)l(h)代表物品在某一时间段内的流行度。h代表某个具体的时间片。

定义8(物品的评分与流行度之间的关系)$f_i(x)$代表物品得到的评分与物品在某一时间段内的流行度之间存在的正相关关系。其中的x就是定义7中的l(h)。

3.2 BiasSVD算法

RMSE是根号下预测值与真实值之间差的平方与总预测次数N的比值。常被用来衡量协同过滤算法的好坏。FunkSVD的核心思想是实现RMSE在训练集上最小。

FunkSVD预测函数为：

$$\hat{r}_{u,i} = q_{i,k}^T p_{u,k} \tag{5}$$

其中，$\hat{r}_{u,i}$代表预测评分。BiasSVD算法在FunkSVD算法的基础上添加了用户偏置参数和物品偏置参数。在下文中将使用u来代表全局平均分，u代表某个数据集上用户的平均评分标准。再将定义4和定义5中的参数加入之后，得到BiasSVD预测函数为：

$$\hat{r}_{u,i} = u + b_u + b_i + q_{i,k}^T p_{u,k} \tag{6}$$

为了得到向量p、q和参数b_u、b_i，我们使用梯度下降算法最小化如下损失函数：

$$\min \sum_{i,j} \left(r_{u,i} - \hat{r}_{u,i}\right)^2 \tag{7}$$

如果只用公式(7)来进行算法处理，又经常会遇到过拟合的问题。所以通常使用正则化方法来对参数b_u、b_i、q_j、p_i进行惩罚，损失函数公式最终如下：

$$\min \sum_{i,j} \left(r_{u,i} - \hat{r}_{u,i}\right)^2 + \lambda(\|p_i\|^2 + \|q_i\|^2 + \|b_u\|^2 + \|b_i\|^2) \tag{8}$$

BiasSVD中存在一个b_u参数用来代表用户本身的偏置因素。而用户的性格和兴趣是b_u的重要组成部分。通过前面的介绍可以知道用户的性格和兴趣会随时间的增长而发生变化。所以可以将b_u参数融入时间信息，来作为实现Time-BiasSVD算法中的一部分。

3.3 时间信息融入用户偏置项

用户的评分习惯会随时间的推移而发生变化，以用户对电影的评分为例。

若干年前某用户观看了一部电影，这部电影给用户产生了一个刺激，这个刺激给用户造成的影响反映为用户对这部的电影的评分。但是随着时间的推移，以下几种原因会改变用户对某类电影的评分：第一，用户看的电影变多导致用户阅历增加；第二，同类型的电影看得太多导致对这类电影的新鲜度降低；第三，经历的事情变多导致用户的兴趣发生变化，原来感兴趣的类型现在可不感兴趣了，反之亦然。

这些因素都会影响用户的评分习惯，最可能的表现是用户对电影进行打分的时候更加严苛。

在生态学中可以使用“S”型曲线[13]来对种群数量变化进行解释。即某种生物数量的增长随着时间的递增并不是呈J型增长，而是S型增长。因为当种群密度增长到某个数值之后，会遭受一系列客观因素的制约，导致其增长趋势受到制约，符合S型增长趋势。

使用“S型”曲线解释用户对电影的评分也是合理的：从现在的时间向前推，越接近评分记录产生的时间，这部电影给用户带来的刺激越大，但用户对此类电影的评分也不可能是呈现J型增长。因为在给用户推荐物品的时间点与评分记录产生的时间点之间，用户也会受到各种其他刺激的影响。

同时，依据心理学中提出的观点，以及郭新明等人的相关研究[14]发现，用户的遗忘规律不会完全依照线性变化方式。用户对于新产生的兴趣在其产生的初期关注度会很高，具有极小的立即遗忘可能性；当用户度过这段时期后，对该兴趣的关注度就会随着时间的延伸而迅速降低，当关注度降低到一定程度时，用户重新回忆起这个兴趣的可能性就很小，对应于该兴趣的遗忘速度，就是表现出速率迅速减小的特点。

综上所述：本文使用变型的Sigmoid函数来描述这个变化趋势。变型的Sigmoid函数在一开始表现出来的特点是，数值一开始维持较高水平且几乎不会递减；在后期，迅速降低随后逐渐变得平缓并最终趋于0。

Sigmoid函数公式为：

$$S(t) = \frac{1}{1+\mathrm{e}^{(-t)}} \tag{9}$$

函数图像如图1所示。

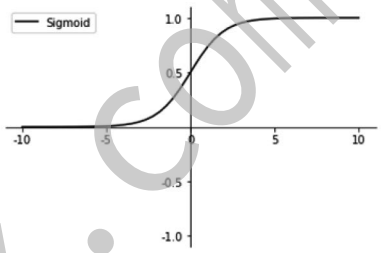

图1 Sigmoid函数图像

Fig.1 Sigmoid function image

Sigmoid函数有很多非常好的性质：它的定义域在$(-\infty, +\infty)$，值域在(0,1)，并且该函数在定义域内为光滑可导函数。

直接使用Sigmoid函数并不能满足我们的要求，因为本文需要的是电影给用户造成的刺激在第一象限内随时间跨度的增加呈现出S型递减趋势。因此本文把Sigmoid函数稍微做一下变型。Time-BiasSVD算法的变化函数$f_u(t)$公式如下：

$$f_u(t) = 1 - \frac{1}{1+e^{\lambda_u - \alpha_u t}} \tag{10}$$

将公式(9)对x轴做对称，之后向y轴正方向平移一个单位，再把整个函数图像向右平移λ_u个单位就可以得到公式(10)。

由于每个用户都是不同的，所以每个用户都有一个属于自己的λ_u和α_u。λ_u可以决定S型函数从哪个点开始下降。α_u决定S型函数下降部分的斜率，代表用户单位时间内兴趣变化的多少。

这两个参数都会在梯度下降算法中不断的迭代更新。t代表用户评分记录中某条评分记录的时间戳，可以直接从数据集中获得。

经过变化之后，$f_u(t)$的一个实例图像如图2所示。

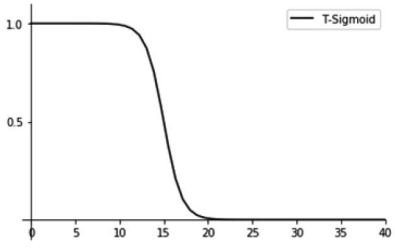

图2 T-Sigmoid函数图像

Fig.2 T-Sigmoid function image

经过以上的描述，可以构造出本文需要的$b_u(t)$，其公式如下：

$$b_u(t) = b_u * f_u(t) \tag{11}$$

$b_u(t)$使得b_u的变化趋势与$f_u(t)$的变化趋势保持一致，且其定义域为$[0,+\infty)$，由于b_u可能为0，所以其值域为$[0,b_u]$。

通过使用公式(11)，可以将原BiasSVD算法的预测函数修改为：

$$\hat{r}_{u,i} = u + b_u(t) + b_i + q_{i,k}^T p_{u,k} \tag{12}$$

3.4 时间信息融入物品偏置项

物品偏置项也是和时间信息有关的量。举例来说，周星驰的电影大话西游之大圣娶亲在刚上映的时候获得的评分并不高。但后来这部电影又重新被拿出来播放，大家给予的评价都非常高。因为这部电影描述的爱情故事正是现代人所崇尚的，也就是说一部电影它讲述的故事和表达的观点在不同的时间段内被人们接受的程度是不一样的，这部电影的流行度在不同的时间段也是不一样的。本文认为某段时间电影的流行度会对电影的评分产生非常大的影响。

下面将对上述观点进行实例分析，以提前设置好的时间段参数t_p为时间片对MovieLens数据集进行划分，t_p可以根据实际情况进行选择和设定。选定了t_p以后就需要计算每个时间片内每部电影的平均评分和流行度。

电影i的平均分$s_i(h)$计算公式如下：

$$s_i(h) = \frac{sum_i(h)}{count_i(h)} \tag{13}$$

其中，$sum_i(h)$代表第h个时间片内电影i获得的所有评分总和，$count_i(h)$代表第h个时间片内某部电影评分记录总数。

电影i的流行度$l_i(h)$计算公式如下：

$$l_i(h) = \frac{m_i(h)}{total(h)} \tag{14}$$

其中，$m_i(h)$代表电影i在第h个时间片内被评价的次数，$total(h)$代表这段时间内所有电影被评价的总次数。

经过对数据集中所有电影的评分与流行度进行数据分析与绘图，可以得出结论：电影获得的评分与电影的流行度呈现正相关。

本文设置时间段参数t_p为1周对数据集进行划分并以id为1的电影在不同时间片内的流行度和平均分制表，得到如表1所示。

表1 平均分流行度数值变化表

Tab.1 Variations of average scores and popularity

时间片序号	电影平均评分	电影流行度	电影评价总数
1	4.375	0.006206	24
2	3.727	0.004811	11
3	4.133	0.005450	15
4	4.100	0.005285	10
⋮	⋮	⋮	⋮
28	3.808	0.003711	26
29	4.000	0.002922	5
30	4.100	0.005543	10

由于评价数量过少的记录不具备可靠性，因此将表一中电影评价总数小于10的行删除。

使用平均分一列和流行度一列做折线图，为了便于观察将流行度一列扩大1000倍。折线如图3所示。

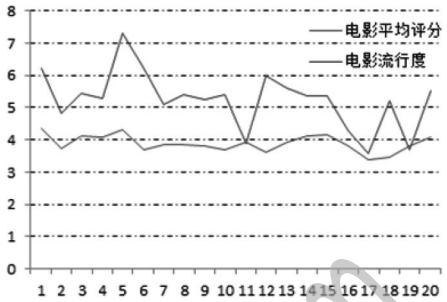

图3 平均分与流行度变化函数图像

Fig.3 Average score and popularity change function image

图3中横坐标代表第h个时间片，纵坐标代表电影在第h个时间片的流行度。从图3可以看出，电影平均评分的曲线和电影流行度的曲线，变化趋势相似度非常高。通过图3，可以得到如下函数：

$$f_i(h) = al_i(h) + b \tag{15}$$

其中，$f_i(h)$代表电影i在第h个时间片的流行度带来的评分。a和b代表是两个参数，在算法中迭代更新。$l(h)$表示电影在第h个时间片的流行度。

但事实上，现实中也确实存在某部烂片由于大量炒作导致电影流行度很高，但实际评分很低的情况。本文给出的解决方案是设置合适的时间段参数t_p，将t_p适当的延长一些。这种炒作出来的烂片短暂的火暴过后，由于实际电影水平很差会导致流行度大大下降。所以只要选择一个合适的t_p就可以将电影实际应有的流行度反映出来。

在公式(6)中有物品偏置项参数b_i可以用来代表电影i所有方面的总偏置数值。在本文中将b_i中与流行度相关的部分单独拆出来融入到用户预测评分函数中。

根据上文的描述将融合时间信息的物品偏置项参数$b_i(h)$设置为：

$$b_i(h) = b_i + f_i(h) \tag{16}$$

经过3.3节与3.4节的介绍，本文最终得出了Time-BiasSVD算法的预测函数为：

$$\hat{r}_{u,i} = u + b_u(t) + b_i(h) + q_{i,k}^T p_{u,k} \tag{17}$$

4 实验结果及分析(Experiment results and analysis)

本节首先介绍实验所用数据集，然后说明评价标准及对比算法，最后给出Time-BiasSVD算法与其他算法的对比实验结果，并对实验结果进行分析。

4.1 数据集及试验环境

本文使用MovieLens数据集进行结果验证。

MovieLens数据集主要包含三个版本，小规模数据集有十万条数据；中等规模数据集有100余万条数据；大规模的数据集过大，由于机器配置问题暂时无法考虑。

通过对中等规模和小规模的MovieLens数据集进行数据分析。发现中等规模的MovieLens数据集上每个用户的所有评分

记录多集中于两小时内。因此，本文猜测获取该数据集的方式为直接对用户进行采访，一名用户一次性对多部电影进行评分。

由于本文的算法需要在较长时间跨度上描述用户兴趣变化趋势和物品流行度。所以中等规模的MovieLens数据集并不适合用来测试本文的Time−BiasSVD算法。因此本文选用拥有十万条数据的MovieLens数据集进行试验验证。

数据集中包含了1997年9月19日到1998年4月22日的943个用户对1682部电影的评分记录。每个用户至少评价了20部电影。评分数据集中有四个属性列，分别是用户ID、电影ID、评分值和时间戳。评分为1—5的整数。

本文根据实际生活中对推荐算法的应用，使用时间点对数据集进行划分。使用历史数据，对未来的数据进行预测。使用时间戳891368000对数据集进行划分，训练集数据72026条，测试集7974条。

4.2 评价指标

本文采用RMSE作为评价算法的标准。推荐算法整体的RMSE越小，意味着推荐算法的推荐质量越高。假设算法对N个产品预测的评分向量表示为$\{\hat{r}_1,\hat{r}_2,\hat{r}_3\ldots\hat{r}_n\}$，这$N$个产品的实际得分向量为$\{r_1,r_2,r_3\ldots r_n\}$，RMSE的计算公式为：

$$\mathrm{RMSE}=\sqrt{\frac{\sum_{i=1}^{N}(r_i-\hat{r}_i)^2}{N}} \tag{18}$$

4.3 参数设置

本文选取了四种算法作为对比算法：

(1)隐语义模型FunkSVD算法，没有考虑用户或者项目本身具有的偏置项因素，也没有将时间信息考虑到算法中。

(2)加入偏置项的隐语义模型BiasSVD算法，将用户偏置项和项目偏置项因素进行了考虑，但是没有考虑时间因素对评分行为的影响。

(3)SVD++算法加入了隐式数据描述的用户对物品的偏好，但是也没有考虑时间因素对用户行为的影响。

(4)Koren等人的timeSVD++算法。该算法也是在SVD++算法的基础上分别在用户偏置项与物品偏置项中融入了时间信息。

在实验中，首先进行特征处理，在原始的评分数据集的基础上根据公式(13)进行流行度计算，为每部电影增加一个流行度属性，方便下文计算。本文将参数P、Q和矩阵b_u、b_i等参数的初始值通过均值为0的正态分布随机抽取获得。在训练集上迭代时，使用批量梯度下降算法不断更新这些参数值，直到函数收敛为止。设置最大迭代次数3000次，跳出迭代条件为本轮RMSE与上轮RMSE之间差值小于1e−5。

在实验中，对每个算法选取好合适的参数后，再将P、Q矩阵中的隐含特征数量K分别设定为10、20、50。并在每个K值下进行多次实验，取平均值为最后的RMSE结果。

4.4 实验结果与分析

实验比较了五种算法在不同隐含特征参数下的结果。在实验中分别设定隐含特征数量为10、20、50。表2给出了不同算法在不同隐含特征参数下RMSE的结果，根据表2可以得出以下结论：

(1)随着隐含特征的增大，各个算法精度都有一定的提高。

(2)BiasSVD算法与FunkSVD算法相比，结果有了不小的提高，这说明在算法中融合偏置因素对结果的提升是有作用的。

(3)本文设计的Time−BiasSVD算法相比BiasSVD算法和FunkSVD算法，以及SVD++算法在结果上都有较大的提升，这说明本文提出的融合时间信息的方法可以有效的提高推荐精度。

(4)本文设计的Time−BiasSVD算法与Koren等人的timeSVD++算法相比，在推荐精度上也有较大提高。

表2 不同算法RMSE比较结果

Tab.2 Comparison of RMSE results among different algorithms

RMSE	K=10	K=20	K=50
SVD++	1.11379	1.11293	1.11041
FunkSVD	1.09976	1.09948	1.09918
BiasSVD	1.09451	1.09266	1.09189
timeSVD++	1.10562	1.10324	1.10087
Time−BiasSVD	1.09182	1.09091	1.09047

5 结论(Conclusion)

本文使用用户对电影的评分信息，以及给予评分的时间信息，设计出了融合时间信息的协同过滤算法Time−BiasSVD。使用变形的Sigmoid函数将时间信息融入到BiasSVD算法的用户偏置项中，同时使用流行度函数将时间信息融入到了物品偏置项中。经过实验结果对比，本文提出的Time−BiasSVD算法与SVD++、FunkSVD，以及工业界常用的传统推荐算法，例如传统隐语义模型算法和koren等人的timeSVD++算法相比，推荐精度有了显著提高。

由于影响推荐效果的因素还有很多，在算法中考虑的因素仍然有丰富的空间。为了更符合用户的动态需求，而不是他过去的需求，可以考虑一些影响用户未来需求的因素：比如潮流的因素，用户所在社交网络的环境特点，用户的消费水平动态变化等。在接下来的研究里，我们将继续挖掘偏置项的特点对算法进行改进，寻找更加完善的模型，以便能够快速智能地得出个性化的推荐方案，有效地保证甚至提高推荐系统的推荐精度。

参考文献(References)

[1] 刘青文.基于协同过滤的推荐算法研究[D].合肥:中国科学技术大学,2013.

[2] 孔维梁.协同过滤推荐系统关键问题研究[D].武汉:华中师范大学,2013.

[3] 王明佳,韩景倜.基于用户对项目属性偏好的协同过滤算法[J].计算机工程与应用,2017,53(6):106−110.

(下转第12页)