文章编号：0253-374X(2026)01-0138-12
DOI：10.11908/j.issn.0253-374x.24296

基于自适应惩罚因子和用户评分行为的协同过滤算法

赵晓群，李煜堃，黄新林
（同济大学 电子与信息工程学院，上海 200092）

摘要：针对传统协同过滤算法头部效应和推荐精度低2个问题，提出了一种基于自适应惩罚因子和用户评分行为的协同过滤算法，惩罚因子通过对热门项目进行惩罚修正了相似度的计算，自适应参数通过遍历的方式找寻不同数据集下的最优惩罚力度，有效缓解了传统算法的头部效应问题；用户评分行为通过考虑用户的评分时间差以及评分的分布特性差异，细化相似度的计算，提高了算法精度。采用4个公开数据集验证改进算法的效果，对于表现较好的MovieLens 1M数据集，保持推荐数目不变时，本文算法F_1分数相比传统算法平均提高约13.9%，有效提高了算法的推荐质量。同时，采用的倒排表构建项目—用户交互矩阵，有效提高了算法运算速度，在MovieLens 1M数据集下，较传统算法运行时间减少约72.1%。

关键词：推荐系统；协同过滤；惩罚因子；自适应参数；用户评分行为
中图分类号：TP301.6　　**文献标志码**：A

An Improved Collaborative Filtering Recommendation Algorithm Based on Adaptive Penalty Factors and User Rating Behavior

ZHAO Xiaoqun, LI Yukun, Huang Xinlin
(College of Electronic and Information Engineering, Tongji University, Shanghai 200092, China)

Abstract: To address the two key issues of head effect and low recommendation accuracy in traditional collaborative filtering algorithms, this paper proposes a collaborative filtering algorithm based on an adaptive penalty factor and user rating behaviors. The penalty factor adjusts similarity calculations by penalizing popular items, while the adaptive parameter identifies the optimal penalty intensity for different datasets through iterative searching, effectively mitigating the long-tail effect of traditional algorithms. User rating behavior further refines similarity calculations by considering time differences between ratings and variations in rating distribution characteristics, thereby improving algorithm accuracy. The proposed algorithm was validated on four public datasets. For the well-performing MovieLens 1M dataset, the algorithm achieved an average F_1 score improvement of approximately 13.9% over traditional algorithms while maintaining the same recommendation count, significantly enhancing recommendation quality. Additionally, the inverted index was employed to construct the item-user interaction matrix, improving computational efficiency. On the MovieLens 1M dataset, the runtime of the proposed algorithm was reduced by approximately 72.1% compared to traditional methods.

Keywords: recommendation system; collaborative filtering; penalty factor; adaptive parameter; user rating behavior

随着人民生活水平的提高和科学技术的不断发展，互联网为人们提供了一个万能的平台，但严重的信息过载问题也接踵而至，用户在海量信息中选出自己感兴趣的内容成为了一个难题[1]。当前，针对信息过载的问题常见的解决方案是使用以推荐系统为核心的信息过滤技术，通过分析用户以往的行为数据，构建模型来识别用户需求和隐含偏好，达到在庞大的信息流中为目标用户挑选合适信息的目的。推荐系统的应用十分广泛，除了熟知的电商以及短视频平台，还涉及到了新闻[2]、音乐[3]等多个领域。

收稿日期：2024-09-11
基金项目：国家重点研发计划(2022YFB3305801)
第一作者：赵晓群，教授，博士生导师，工学博士，主要研究方向为通信与信息理论。E-mail：zhao_xiaoqun@tongji.edu.cn
通信作者：李煜堃，工学硕士，主要研究方向为智能推荐技术。E-mail：459600596@qq.com

论文拓展介绍

在大部分推荐系统使用的推荐算法中，协同过滤算法以其实现简单、适应性强以及无需明确特征描述等特点成为最常用的算法之一。但传统协同过滤算法因为其相似度计算过于简单的缘故存在两大严重问题，即头部效应问题和推荐精度低问题。

对于上述所提两大问题，国内外学者们已进行大量研究并取得显著成果。针对头部效应问题，主要通过对热门项目进行惩罚的方式来修正相似度的计算。刘超慧等[4-5]在相似度计算中引入了热门惩罚因子，通过削减热门物品对相似度计算的影响，提高了相似度的计算精度，降低了算法的平均绝对误差。马汉达等[6]提出了一种基于左归一化图神经网络的推荐模型，令卷积网络中的每层卷积层归一化约束逐层加强，实现了热门衰减的效果，通过实验在3个数据集中证实算法推荐精度有效提高。邹洋等[7]将万有引力公式引入社交网络中，提出了一种基于多权重相似度的随机漫步推荐算法，考虑到了社交网络以及热门项目的影响，有效提高了算法的召回率和精确率。张卫国等[8]将注意力机制融入基于项目的协同过滤算法中，提出了一种融合深度去噪自编码器和注意力机制的推荐算法，通过惩罚活跃用户减少对实验的影响，并在2个公开数据集上验证了有效性，显著提高了算法的推荐性能。段琼等[9]在传统协同过滤中加入了商品流行度概念，对热门的流行商品进行惩罚，并且提出了融合商品流行度与信任度的混合推荐算法，有效降低了算法的均方根误差。张洁等[10]在Pearson相似度公式中加入用户惩罚因子，将用户间的头部效应修正，并与评分信息熵进行融合，形成了一种相似度计算的新方法，有效提高了推荐质量，缓解了传统算法的头部效应。上述所提研究现状中，虽然都通过对热门项目进行惩罚的方式缓解了协同过滤的头部效应问题，但并没有考虑到惩罚的力度，即对于不同数据集及不同算法输入而言，都应该有一个最佳的惩罚力度而不是任意惩罚。同时，对于热门项目也应该有一个定位，即什么样的项目属于热门项目，而不是对所有项目都按照与交互量成反比的方式进行统一惩罚。

针对推荐精度低问题，主要通过挖掘更深层次的数据关联来细化相似度的计算。一些研究人员用辅助信息如用户信息和项目信息等来帮助推荐[11]，Yu等[12]提出了MHCN网络，通过用户之间的高阶语义关系来提升社交推荐效果。Yu等[13]提出了BEST-CF模型，采用最优组合预测的策略，构建了优化模型，并通过约束条件下的求解来确定优化目标函数，从而获取组合预测的加权系数。Wang等[14]提出了一种基于视觉循环卷积矩阵分解的电影推荐系统，该系统通过预训练的卷积神经网络从电影海报或剧照等图像中提取视觉特征，以补充项目描述文本。此外，还通过引入用户偏好的置信机制来优化损失函数，进而提升推荐的准确性。Tran等[15]通过解耦表示更好地捕捉了由多个隐藏因素影响的用户偏好，从而解决了用户与项目交互的稀疏性问题，并将无法解释的维度映射到文字。Zhou等[16]认为构建动态项目—项目图是多余的，并主张在模型训练前冻结该图，以实现有效且高效的推荐，尽管该想法没有充分利用丰富的多模态信息。Zou等[17]通过跳跃关系将用户与用户、用户与项目联系起来，更加充分地描述用户行为特性，并使用TSPR算法生成粗略的推荐列表。Xu等[18]通过分解信任矩阵，获得了用户的信任特征和受信任特征，将无向信任转变为定向信任，改进后的TrustSVD算法产生了全新的PMFTrustSVD算法。Weng等[19]通过灰色关联分析改进了用户相似度计算，构建了包含隐性社交关系的SRGRA算法，提升了推荐质量。Salamat等[20]提出的HeteroGraphRec模型，通过综合用户社交关系、互动历史和商品相似性数据，深化了用户偏好分析。杨北辰等[21]提出了一种基于物品预测得分与基于用户偏好得分的两阶段联合推荐算法，引入时间权重系数和用户评分偏好因子改进了用户相似度的计算，降低了算法的MAE。张婷婷等[22]利用双神经网络结构，学习预期奖励并考虑邻居协同作用，提出了一种神经Bandits推荐算法COEENet，该算法融合了协同过滤思想，构造决策器进行最终决策，有效解决了传统协同过滤的数据稀疏问题，提高了算法的推荐精度。黄晓宇等[23]提出了一种基于“少数派报告”的协同过滤模型，设计了能自适应匹配预测目标训练数据的选择算法，并且采用Elastic Net模型缓解数据稀疏问题，有效降低了预测误差。李驰等[24]提出了一种基于解耦图卷积网络的协同过滤推荐模型，利用负采样为初始向量注入图结构特征并使用逆倾向分数加权的损失函数训练模型参数，在缓解数据稀疏的基础上有效降低了算法的训练效率。姬璐等[25]将人的兴趣与评分差异融入协同过滤推荐算法，并利用信息熵和最大评分值分别度量，有效提高了算法的推荐精度。朱金侠等[26]提出了一种融合图卷积注意力机制的协同过滤推荐模型，考虑了邻居节点在传播聚合时的重要性，使用户与项目的嵌入向量表达更加合理化，有效提高了模型的泛

化能力及推荐精度。上述所提研究现状中,虽然都通过对数据的进一步挖掘利用有效提高了算法的推荐精度,但并没有很好地考虑到用户评分行为,即用户评分时间及评分值分布特性的综合考量。考虑用户评分时间能有效量化用户的兴趣变化,而考虑用户评分值的分布特性能进一步挖掘用户间的相似关系。

针对上述提出的两大问题以及研究现状中的不足,本文提出一种基于自适应惩罚因子和用户评分行为的改进协同过滤算法(An Improved Collaborative Filtering Recommendation Algorithm Based on Adaptive Penalty Factors and User Rating Behavior,CFAU),自适应参数找寻不同数据集及算法输入下的最优惩罚力度,更好地缓解协同过滤的头部效应;用户评分行为的考虑细化了相似度的计算,进一步提高算法计算精度。在实验验证方面,采用 MovieLens 1M、MovieLens Latest、Amazon Clothing 和Amazon Shoes共4个数据集进行实验仿真,验证了本文算法的推荐精度及泛化性。

1 自适应惩罚因子

协同过滤的头部效应是指,在推荐系统中,热门项目会受到过度推荐,而一些冷门项目则会被忽视,最终导致热门项目不断被推荐,并在数据更新后越来越热门,冷门项目则越来越冷门,难以被推荐,直接导致推荐结果的单一和失真。这便是协同过滤的一个重大问题,即处理稀疏数据能力弱、头部效应显著。为解决该问题,不少学者使用了“热门惩罚”的方式来惩罚热门项目,但是并没有很好地考虑到热门项目的区分度。由此缺陷,设计了自适应惩罚因子如下。

定义某一项目i的项目热门度P_i:在用户—项目交互数据集中,评价过该项目的用户数与总评价条数的比值即为该项目的项目热门度,如式(1)所示:

$$P_i=\frac{\sum_{k=1}^{N}\lambda_{k,i}}{N} \tag{1}$$

式中:P_i为项目热门度,$P_i\in[0,1]$,其值越大代表该项目越热门;N为数据总条数;$\lambda_{k,i}$为第k条数据是否是对于项目i的评分数据,若是,则$\lambda_{k,i}=1$,否则$\lambda_{k,i}=0$。

为了减小热门项目对相似度计算做出过度贡献而导致的头部效应,在此设计一个自适应惩罚因子来抑制热门项目产生的影响。因子具体表达式为

$$\omega_i=\begin{cases}1, H_i<T\\ \dfrac{1}{1+(N_i-NT)^{\alpha}}, H_i\geqslant T\end{cases} \tag{2}$$

式中:ω_i为项目i的自适应惩罚因子,$\omega_i\in[0,1]$,值越小代表对热门项目i的惩罚力度越大,对相似度的贡献越低;T为项目热门度阈值;N_i为项目i在数据集中被交互的数据条数;α为自适应参数,$\alpha>0$,表示对热门项目的惩罚力度;H_i为项目i的项目热门度,$H_i\in[0,1]$,值越大代表项目i越热门。

项目热门度阈值T的设置是为了防止对每个项目都进行惩罚而导致结果失真。当$H_i<T$时,该项目会被判定为非热门项目,不需要进行惩罚,对于相似度计算的贡献按“1”处理;当$H_i\geqslant T$时,该项目会被判定为热门项目,利用因子对其进行加权制约调整,以减少其在相似度计算中的比重。

α的设置是为了让该算法可以在任何输入下都能通过遍历的方式找寻此时的最优热门惩罚力度。例如,在某一数据集下,若该数据集的数据比较稀疏,且输入的近邻人数及推荐个数较小,那么此时对热门项目进行惩罚会适得其反,当$\alpha=0$或一个很小的值才会有一个较好的结果。若该数据集较稠密,且输入的近邻人数和推荐个数较大,那么对于热门项目的惩罚就很有必要,当$\alpha=1$或更大时才会获得一个较好的结果。而具体的惩罚力度可以通过在一个预先给定的范围$[\alpha_l,\alpha_r]$内以步长0.1进行遍历搜索,比较当前算法指标值F_1-s是否变得更好,若是则更新当前最优推荐结果及最优指标,若不是则不更新,最终输出该算法最优推荐结果。

2 用户评分行为

协同过滤除头部效应外,另一个严重问题是推荐精度低,即由于数据稀疏问题,导致相似度计算不准确或无法计算。为解决该问题,可以通过进一步挖掘现有数据如使用大模型,成功提高了算法的推荐精度,但并没有考虑到用户自己独特的评分行为。用户评分行为可以拆分为用户评分时间和用户评分偏好,故从这2个方面入手进行算法改进。

2.1 用户时间因子

在讲究时效性的推荐系统中,时间是一个很关键的数据信息。用户兴趣往往跟记忆挂钩,记忆又与时间成反比,即随着时间流逝,在不加以复习的情况下,用户对某一项目的记忆会越来越少直至遗忘。

在相似度计算时，为了将用户的兴趣因素考虑在内，参考艾宾浩斯遗忘曲线设计了用户时间因子$R(t)$为

$$f(t)=ae^{-bt}+c \tag{3}$$

$$R(t)=\frac{e^{\frac{1}{\ln(1+t)}}-e^{\frac{-1}{\ln(1+t)}}}{e^{\frac{1}{\ln(1+t)}}+e^{\frac{-1}{\ln(1+t)}}} \tag{4}$$

式中：a、b、c为遗忘曲线的参数，代表每个人不同的遗忘速率和基准，因人而异，$R(t)\in[0,1]$，值越大代表用户的评分时间t越新，即兴趣越新，对相似度贡献越高。函数对比曲线如图1所示。

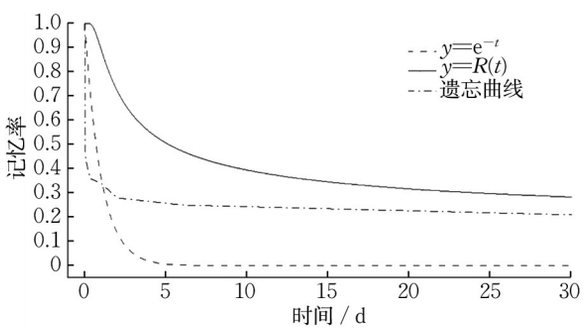

图 1　函数对比曲线

Fig. 1　Comparison of functions

由图1可见，在函数$y=e^{-t}$的基础上加参调节形成了艾宾浩斯遗忘曲线，但两函数均在短时间内大幅衰减，并在时间长后缓慢衰减，这与用户的实际兴趣衰减趋势有所偏差。按照两函数曲线走势分析，一天内用户的兴趣会衰减70%以上，这显然是不现实的，所以要将其衰减速度放缓，这里参考了双曲正切函数，并对其进行变形，形成了上述的用户时间因子，使其更符合用户的实际兴趣衰减趋势。

2.2　用户评分偏好

用户评分偏好代表用户个人的评分习惯，而传统余弦相似度并没有考虑到这一情况，只是进行单纯的向量数值计算，导致推荐精度不高。由此科学家提出了Pearson相关系数，通过引入均值的概念，提高相似度的计算精度，但仍没有考虑到用户评分的分布特性。

用户评分的分布特性可以由用户历史评分数据的偏度和峰度共同决定。偏度表示用户评分的分布形态，即偏向于打高分、打低分还是打平均分，峰度表示用户评分的集中程度，即评分是否集中在某个分数。偏度和峰度的表达式为

$$S=\frac{N}{(N-1)(N-2)}\sum_{i=1}^{N}\left(\frac{x_i-\bar{x}}{\sigma}\right)^3 \tag{5}$$

$$K=\frac{N(N+1)}{(N-1)(N-2)(N-3)}\sum_{i=1}^{N}\left(\frac{x_i-\bar{x}}{\sigma}\right)^4-\frac{3(N-1)^2}{(N-2)(N-3)} \tag{6}$$

式中：N为数据总条数；x_i为用户对第i个项目的具体评分值；$\bar{x}$为用户的平均评分值；σ为用户的评分标准差。对于2个用户而言，用户间的偏度和峰度差值越小说明其打分的分布特性越接近，设计用户评分偏好因子为

$$A(u,v)=\frac{2N_SN_K}{N_S+N_K} \tag{7}$$

式中：$A(u,v)$为用户u和用户v之间的用户评分偏好因子；N_S和N_K为2个用户间的偏度和峰度的差值归一化处理后的结果，表达式分别为

$$N_S=1-\frac{\Delta S-S_{\min}}{S_{\max}-S_{\min}} \tag{8}$$

$$N_K=1-\frac{\Delta K-K_{\min}}{K_{\max}-K_{\min}} \tag{9}$$

式中：ΔS为用户u、v的偏度之差的绝对值；$S_{\max}$和$S_{\min}$分别为所有用户偏度中的最小值和最大值。

3　CFAU算法描述

基于上述提出的自适应热门惩罚因子和用户评分行为概念，设计CFAU算法相似度计算公式为

$$\mathrm{sim}(u,v)=\frac{\sum_{i\in I_{u,v}}(g_{u,i}-\overline{g_u})(g_{v,i}-\overline{g_v})\omega_iR(\Delta t)}{\sqrt{\sum_{i\in I_u}[(g_{u,i}-\overline{g_u})\omega_i]^2}\sqrt{\sum_{i\in I_v}[(g_{v,i}-\overline{g_v})\omega_i]^2}}A(u,v) \tag{10}$$

$$\Delta t=\left\lfloor\frac{|t_{u,i}-t_{v,i}|}{86\,400}\right\rfloor \tag{11}$$

式中：I_u、I_v分别为用户u、v评分过的项目集合；$I_{u,v}$为两用户共同评分过的项目集合；$g_{u,i}$、$g_{v,i}$分别为用户u、v对项目i的具体评分值；$\overline{g_u}$、$\overline{g_v}$分别为两用户评分的平均分；Δt为两用户对同一项目i评分的时间差的绝对值除以86 400并向下取整，即转化为以天为单位的时间差，认为同一天内的评分参考价值相同。

评分预测公式为

$$Y_{u,i}=\overline{g_u}+\frac{\sum_{v\in U_u}\mathrm{sim}(u,v)(g_{v,i}-\overline{g_v})}{\sum_{v\in U_u}|\mathrm{sim}(u,v)|} \tag{12}$$

式中：U_u为目标用户u的近邻用户集合；i为目标用户u未交互过的项目。

CFAU算法流程图如图2所示。

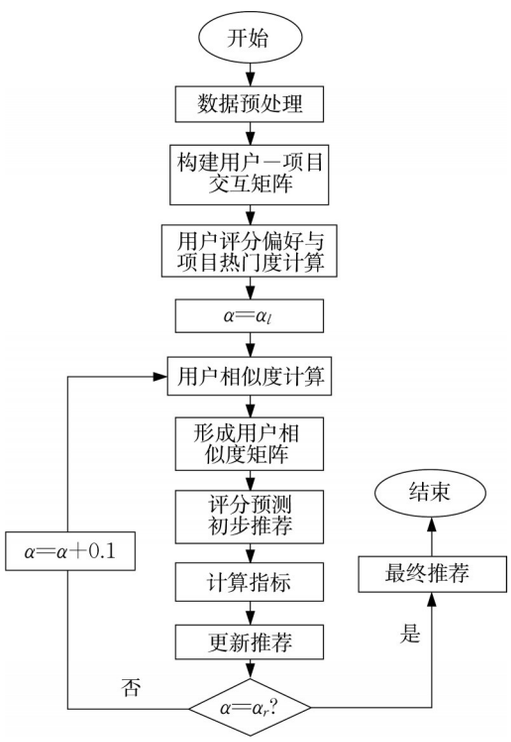

图2 算法流程

Fig. 2 Flowchart of algorithm

为便于理解，以目标用户u为例，列出CFAU算法具体步骤。算法名称：基于自适应惩罚因子和用户评分行为的协同过滤算法CFAU；输入：用户—项目数据集，目标用户u，参数范围$[\alpha_l,\alpha_r]$，近邻人数k，推荐个数n；输出：给目标用户u的最佳项目推荐集合L_m。

步骤为：①数据预处理，读取数据并划分为训练集与测试集。②由训练集构建用户—项目交互矩阵，利用式(1)和式(7)计算每个项目的项目热门度和两两用户间的用户评分偏好因子。③初始化参数$\alpha=\alpha_l$，$F_{1m}=0$，L_m为none。其中F_{1m}代表最佳F_1，L_m代表F_{1m}下的最优推荐结果集合。④利用式(10)计算用户相似度并形成相似度矩阵$\boldsymbol{W}$。⑤利用式(12)进行评分预测生成初步推荐，并计算当前的指标F_1，若$F_1>F_{1m}$，则更新参数$F_{1m}=F_1$，$L_m=L$；否则不变。⑥判断若$\alpha\neq\alpha_r$，则$\alpha=\alpha+0.1$，返回步骤④；否则输出最优推荐结果L_m。

CFAU算法伪代码如下：

输入：用户—项目数据集，目标用户u，参数范围$[\alpha_l,\alpha_r]$，近邻人数k，推荐个数n

输出：给目标用户u的最佳项目推荐集合L_m

1. 数据预处理，构建用户—项目交互矩阵D
2. for i, x_i, t_i in $D_1.values()$ do
3. $P_i\leftarrow\frac{\sum_{k=1}^{N}\lambda_{k,i}}{N}$ //计算每个项目i的热门度
4. $A(u,v)\leftarrow\frac{2N_SN_K}{N_S+N_K}$ //计算两两用户间的评分偏好因子
5. end for
6. $\alpha\leftarrow\alpha_l, F_{1m}\leftarrow 0, L_m\leftarrow None, d=\{\}$ //初始化参数
7. for $u, item$ in $D.items()$ do
8. for i, g_i, t in item do
9. $d[i]\leftarrow u, g_i, t$ //构建家电-项目倒排表
10. end for
11. end for
12. for $\alpha\leftarrow\alpha_l$ to α_r by 0.1 do
13. $$sim(u,v)\leftarrow\frac{\sum_{i\in I_{u,v}}(g_{u,i}-\overline{g_u})*(g_{v,i}-\overline{g_v})*\omega_i*R(\triangle t)}{\sqrt{\sum_{i\in I_u}[(g_{u,i}-\overline{g_u})*\omega_i]^2}\sqrt{\sum_{i\in I_v}[(g_{v,i}-\overline{g_v})*\omega_i]^2}}A(u,v)$$ //计算相似度
14. $W\leftarrow sim(u,v)$ //生成相似度矩阵
15. $U_u=sorted(W[u].items(), key=lambda\ x: x[1], reverse=Ture)[:k])[:k]$ //近邻用户集合
16. for v in U_u do
17. $Y_{u,i}\leftarrow\overline{g_u}+\frac{\sum_{v\in U_u}sim(u,v)*(g_{v,i}-\overline{g_v})}{\sum_{v\in U_u}|sim(u,v)|}$ //预测用户u未评分过的项目i的评分
18. end for
19. $L\leftarrow n, Y_{u,i}$ //由预测评分最高的前n个项目形成初步推荐，并计算当前指标
20. If $F_1>F_{1m}$ do
21. $F_{1m}\leftarrow F_1, L_m\leftarrow L$ //更新参数
22. end if
23. end for

24. **return** L_m

4　实验与分析

4.1　实验数据

在实验数据方面，采用MovieLens数据集中的MovieLens 1M和MovieLens Latest的2个数据集，以及Amazon review data数据集中的Amazon Clothing和Amazon Shoes的2个数据集，共4个数据集。其中，MovieLens系列数据集格式相对完善，交互数据只包含用户id、项目id、评分值和评分时间4列，可以直接使用；Amazon review data数据集还额外包含了用户年龄、评价文本等信息，剔除后只保留上述4项内容即可。选取90%数据作为训练集、10%数据作为测试集。4个数据集按上述描述顺序标为D_1、D_2、D_3、D_4，具体信息如表1所示。

表1　实验数据具体信息

Tab. 1　Experimental data

数据集	用户数	项目数	评分数	稀疏度/%
D_1	6 040	3 900	1 000 209	95.75
D_2	610	9 000	100 837	98.16
D_3	73 590	48 410	389 877	99.98
D_4	128 794	66 370	581 933	99.99

4.2　评价指标

评价指标采用推荐系统中常用的F_1分数，它是召回率与精确率的综合加权值，范围在0到1之间，值越大代表算法推荐效果越好，召回率、精确率以及F_1的表达式为

$$F_1=\frac{2P_r\cdot R_e}{P_r+R_e} \tag{13}$$

$$R_e=\frac{\sum_u|R(u)\cap T(u)|}{\sum_u|T(u)|} \tag{14}$$

$$P_r=\frac{\sum_u|R(u)\cap T(u)|}{\sum_u|R(u)|} \tag{15}$$

式中：$R(u)$为对目标用户u推荐的n个项目集合；$T(u)$为测试集中目标用户u感兴趣的项目集合，评分值不小于3。

4.3　算法参数分析

所提CFAU算法有2个参数，分别是项目热门度阈值T和自适应参数α，接下来进行2组实验，分别确定这2个参数的取值并给出分析。

(1)自适应参数α取值范围。α在范围$[\alpha_l,\alpha_r]$内进行遍历，该范围的大小决定了算法的计算精度与效率。过大的范围会导致算法复杂度大幅增加，过小的范围会导致难以找到最优参数值，所以该范围的确定十分重要。在上述4个实验数据集下，设置推荐个数$n=10$，近邻人数$k=15$，项目热门度阈值$T=0$，$[\alpha_l,\alpha_r]$预设为$[0,1.5]$，令α以步长0.1遍历该范围，得到CFAU算法F_1分数随α值变化图如图3所示。由图可知，在4个数据集中，当α分别等于0.4、0.3、0.2、0.1时，算法指标值达到最大，且当$\alpha>0.8$后，算法指标值不再起伏，而是持续下降，表明此时α已经过大，对热门项目产生了“过惩罚”，导致算法整体推荐效果下降。故将α范围$[\alpha_l,\alpha_r]$设置在$[0,0.8]$范围内进行后续实验。

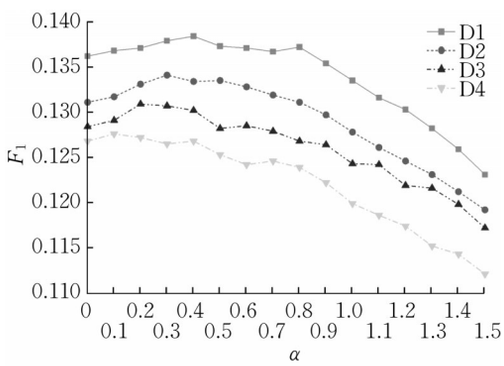

图3　F_1分数值随参数α值变化曲线

Fig. 3　Variation of F_1 score with α

(2)项目热门度阈值T取值。为验证T值对实验结果的影响，在4个实验数据集中，设置推荐个数$n=10$，近邻人数$k=15$，α分别等于0.4、0.3、0.2、0.1，在$[0,0.4]$的范围内以步长0.05进行实验。实验结果如图4所示。由图可知，在4个数据集中，当T分别等于0.15、0.10、0.05、0.05时，算法指标达到最大，且存在明显峰值情况，这是因为当T值过小时，对热门项目划分力度过大，几乎所有项目都被划分为热门项目，从而产生“过惩罚”；当T值过大时，对热门项目划分力度过小，只有极度热门的项目才会被划分为热门项目，从而没有产生适当的惩罚效果。

4.4　算法性能对比分析

确定完CFAU算法参数取值及范围后，现进行协同过滤领域内同类型算法的对比分析，以验证其有效性。对比算法分别为：基于用户的协同过滤算法UCF，文献[5-6]算法CF-1、文献[16]算法CF-2、文献[25]算法CF-3和文献[22]算法CF-4。

CFAU算法参数设置由上述实验所定，即在4个数据集中项目热门度阈值T分别取值0.15、0.10、0.05、0.05，自适应参数范围统一为$[0,0.8]$，在不同

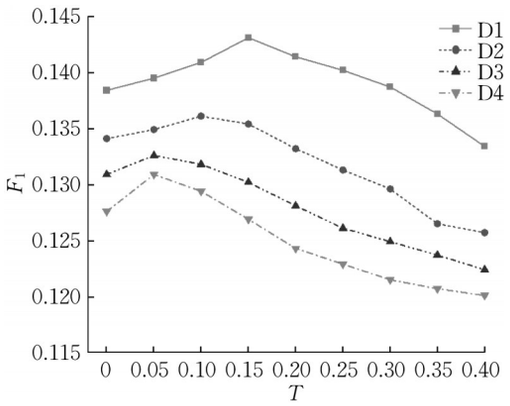

图4 F_1分数值随参数T值变化曲线

Fig. 4 Variation of F_1 score with T

k和不同n下,进行2组对照试验。

(1)保持推荐个数n不变。保持$n=10$,将6种算法分别在近邻人数k=2、3、5、10、15、20、25、30、35、40、45、50时在4个数据集下运行,算法F_1分数随k值变化数据如表2—5所示,算法F_1分数平均值对比如表6所示。对应变化及对比图如图5、图6所示。

(2)保持近邻人数k不变。保持$k=15$,将6种算法分别在推荐个数n=2、3、5、10、15、20、25、30、35、40、45、50时在4个数据集下运行,算法F_1分数随n值变化数据如表7—10所示,算法F_1分数平均值对比如表11所示。对应变化及对比图如图7、图8所示。

表2 D_1中不同k值下算法F_1值

Tab. 2 F_1 scores of algorithm under different k values in D_1

算法名称	k											
	2	3	5	10	15	20	25	30	35	40	45	50
UCF	0.062 5	0.085 7	0.101 2	0.117 7	0.123 8	0.127 6	0.129 6	0.129 6	0.128 4	0.127 3	0.126 8	0.125 3
CF—1	0.059 2	0.080 3	0.098 6	0.120 2	0.129 6	0.127 3	0.130 7	0.128 6	0.126 3	0.128 9	0.130 1	0.128 6
CF—2	0.069 5	0.088 6	0.105 3	0.124 3	0.132 3	0.131 1	0.134 6	0.133 3	0.134 4	0.133 8	0.133 5	0.133 6
CF—3	0.069 9	0.089 4	0.106 1	0.127 8	0.132 6	0.135 9	0.135 2	0.134 5	0.136 1	0.134 4	0.133 9	0.134 2
CF—4	0.070 5	0.096 1	0.113 8	0.133 2	0.139 6	0.142 1	0.141 9	0.141 8	0.140 4	0.140 3	0.139 8	0.139 6
CFAU	0.071 6	0.098 7	0.116 9	0.137 3	0.143 1	0.146 1	0.143 5	0.146 2	0.144 3	0.143 4	0.144 1	0.143 7

表3 D_2中不同k值下算法F_1值

Tab. 3 F_1 scores of algorithm under different k values in D_2

算法名称	k											
	2	3	5	10	15	20	25	30	35	40	45	50
UCF	0.063 4	0.079 4	0.095 4	0.113 0	0.117 9	0.121 5	0.126 0	0.127 0	0.124 0	0.123 9	0.123 1	0.122 7
CF—1	0.057 0	0.075 1	0.092 8	0.113 8	0.120 5	0.122 9	0.123 7	0.123 5	0.125 7	0.127 0	0.124 5	0.125 0
CF—2	0.069 1	0.082 2	0.099 9	0.119 1	0.124 9	0.126 1	0.128 8	0.129 5	0.128 5	0.128 9	0.128 6	0.129 0
CF—3	0.069 5	0.083 6	0.102 2	0.123 1	0.127 4	0.127 9	0.129 8	0.131 4	0.130 4	0.130 5	0.129 7	0.129 6
CF—4	0.070 2	0.086 1	0.105 5	0.125 6	0.132 3	0.134 4	0.135 7	0.137 1	0.135 8	0.137 2	0.135 7	0.135 8
CFAU	0.071 5	0.070 3	0.109 9	0.129 7	0.136 1	0.138 2	0.139 8	0.141 3	0.139 5	0.140 9	0.139 6	0.139 3

表4 D_3中不同k值下算法F_1值

Tab. 4 F_1 scores of algorithm under different k values in D_3

算法名称	k											
	2	3	5	10	15	20	25	30	35	40	45	50
UCF	0.062 2	0.076 3	0.092 7	0.111 4	0.114 8	0.118 6	0.124 5	0.124 9	0.122 1	0.122 3	0.121 4	0.120 6
CF—1	0.056 4	0.073 9	0.089 7	0.110 5	0.115 8	0.120 6	0.121 4	0.121 9	0.123 8	0.125 5	0.123 1	0.123 2
CF—2	0.068 8	0.080 1	0.097 6	0.116 9	0.121 4	0.123 6	0.125 7	0.127 5	0.125 5	0.126 3	0.126 2	0.126 7
CF—3	0.068 9	0.080 6	0.100 2	0.118 3	0.124 8	0.123 9	0.127 1	0.129 8	0.127 5	0.128 6	0.127 6	0.127 3
CF—4	0.069 4	0.081 1	0.101 3	0.121 8	0.128 6	0.130 5	0.132 6	0.134 7	0.133 5	0.135 6	0.133 6	0.133 9
CFAU	0.070 7	0.086 1	0.106 4	0.125 9	0.132 6	0.134 3	0.137 9	0.138 8	0.137 1	0.139 6	0.137 2	0.137 1

由上述2组实验可知,CFAU算法在4个数据集上的表现均优于其他对比算法,且在表现最好的MovieLens 1M数据集中,保持推荐数目$n=10$不变的情况下,CFAU算法F_1分数均值比传统算法UCF提高约13.9%。实验结果表明,本文所提CFAU算法具有较好的有效性和一定的泛化性。

各算法在4个数据集中的性能差异是由数据稀疏性带来的,数据集越稀疏就越难以找寻用户间的相似度,从而导致推荐精度下降。故各算法均在数据最密集的MovieLens 1M上获得最佳表现。

表5　D_4中不同k值下算法F_1值

Tab. 5　F_1 scores of algorithm under different k values in D_4

算法名称	k											
	2	3	5	10	15	20	25	30	35	40	45	50
UCF	0.061 5	0.075 2	0.091 5	0.110 1	0.113 3	0.117 1	0.123 1	0.123 4	0.120 7	0.120 9	0.120 0	0.119 2
CF－1	0.055 8	0.073 0	0.088 5	0.109 1	0.114 2	0.119 2	0.1200	0.120 5	0.122 4	0.124 1	0.121 7	0.121 8
CF－2	0.068 1	0.079 1	0.096 4	0.115 5	0.119 8	0.122 1	0.124 1	0.126 0	0.123 9	0.124 8	0.124 7	0.125 2
CF－3	0.068 2	0.079 5	0.099 0	0.116 8	0.123 3	0.123 7	0.125 6	0.128 3	0.125 9	0.127 1	0.126 1	0.125 8
CF－4	0.068 6	0.079 8	0.099 9	0.120 2	0.126 9	0.128 8	0.131 0	0.133 1	0.131 9	0.133 1	0.132 1	0.132 4
CFAU	0.069 8	0.084 8	0.105 0	0.124 3	0.130 9	0.132 6	0.137 4	0.137 2	0.135 5	0.136 9	0.135 6	0.135 5

表6　不同k值下算法F_1均值对比

Tab. 6　Comparison of algorithm F_1 means under different k values

算法名称	D_1	D_2	D_3	D_4
UCF	0.115 5	0.111 4	0.109 3	0.108 0
CF－1	0.115 7	0.111 0	0.108 8	0.107 5
CF－2	0.121 2	0.116 2	0.113 9	0.112 5
CF－3	0.122 5	0.117 8	0.115 4	0.114 1
CF－4	0.128 3	0.122 6	0.119 7	0.118 2
CFAU	0.131 6	0.126 3	0.123 6	0.122 1

4.5　算法时间对比分析

推荐算法由于要计算大量的用户历史数据，时间复杂度随着数据量的增加会大幅增长。本文所提CFAU算法涉及到额外的用户评分偏好因子计算以及自适应参数的遍历，如果采用跟传统算法一样的计算方式，虽然精度得以提高，但时间复杂度成倍增加，忽略时间复杂度的精度是没有意义的。故本文所提算法在数据处理环节采用了“倒排表”方式，将传统的用户—项目交互矩阵构造为项目—用户交互

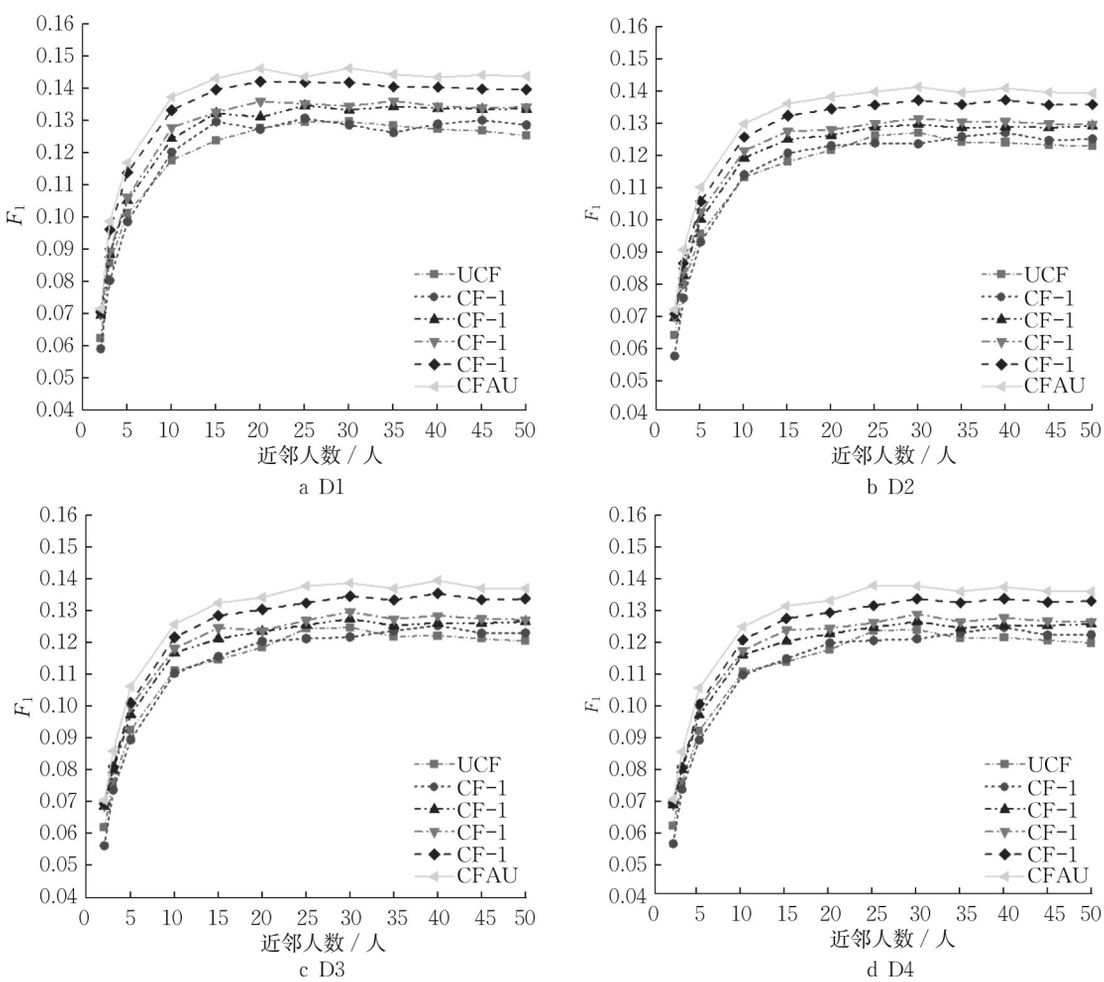

图 5　不同k值下的F_1对比曲线

Fig. 5　Comparison of F_1 under different k values

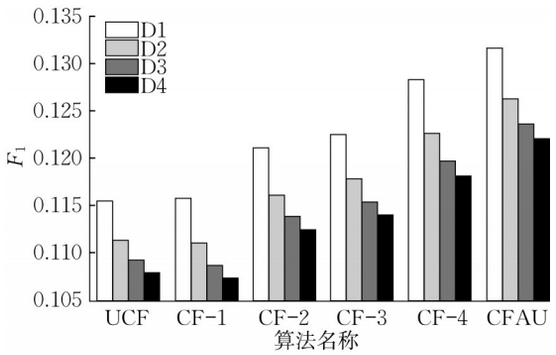

图6 不同k值下的F_1均值对比

Fig. 6 Comparison of F_1 means under different k values

矩阵，即形为：{item1:[user1, user2], item2:[user1, user3], .……}。其中每个user是包含了身份id、对item的评分值和评分时间3个元素的元组。

假设数据集包含n个用户、m个项目、k条评分信息，那么传统算法直接遍历计算两两用户相似度的时间复杂度为$O(n^2\overline{m})\approx O(nk)$，其中$\overline{m}$为每个用户的平均评分项目数，而使用倒排表方式构建数据的算法计算时间复杂度为

$$O(\sum_{i=1}^{m}\frac{\overline{u_i}(\overline{u_i}-1)}{2})\approx O(\frac{k^2}{m})$$

其中$\overline{u_i}$为每个项目的平均评分用户数。由此可知，对于大型数据集，n、m、k都非常大，传统算法的时间复杂度$O(nk)$会非常大；由于$k\ll nm$，故使用倒排表的时间复杂度$O(\frac{k^2}{m})$就相对小很多，优势非常明显，能有效应对用户数量多且评分稀疏的大型数据集。

在4个数据集中使用CFAU算法和传统算法UCF进行了运行时间对比实验，每个数据集下两算法各运行5次，将5次时间的均值作为该数据集下的算法运行时间。对比结果图如图9所示。

由实验结果可知，在4个数据集下，CFAU算法的运行时间均小于UCF算法，且数据集的稀疏性越高，时间节省就越多。表明本文所提倒排表的数据处理方式可以有效应对数据稀疏场景下的相似度计

表7 D_1中不同n值下算法F_1值

Tab. 7 F_1 scores of algorithm under different n values in D_1

算法名称	n											
	2	3	5	10	15	20	25	30	35	40	45	50
UCF	0.059 9	0.078 3	0.100 6	0.123 8	0.134 5	0.140 5	0.138 5	0.136 6	0.135 2	0.130 7	0.126 8	0.122 8
CF−1	0.059 2	0.077 2	0.101 2	0.129 6	0.135 2	0.138 9	0.140 5	0.137 5	0.134 4	0.131 6	0.127 5	0.125 3
CF−2	0.062 2	0.080 6	0.105 1	0.132 3	0.138 5	0.143 9	0.145 3	0.142 4	0.138 7	0.136 3	0.131 8	0.130 2
CF−3	0.063 3	0.082 2	0.106 5	0.132 6	0.139 7	0.145 6	0.146 9	0.144 4	0.139 9	0.138 2	0.133 3	0.132 8
CF−4	0.065 4	0.088 3	0.11 31	0.139 6	0.146 5	0.151 1	0.152 3	0.150 2	0.146 8	0.145 5	0.139 9	0.139 2
CFAU	0.06 97	0.091 3	0.117 1	0.143 1	0.152 1	0.155 7	0.155 4	0.153 5	0.151 2	0.148 3	0.144 1	0.141 6

表8 D_2中不同n值下算法F_1值

Tab. 8 F_1 scores of algorithm under different n values in D_2

算法名称	推荐个数n											
	2	3	5	10	15	20	25	30	35	40	45	50
UCF	0.053 3	0.069 8	0.090 8	0.117 9	0.125 5	0.131 7	0.132 2	0.129 9	0.125 9	0.123 1	0.120 3	0.117 9
CF−1	0.054 1	0.069 8	0.090 3	0.120 5	0.127 8	0.131 9	0.132 4	0.129 4	0.128 2	0.123 9	0.120 3	0.116 9
CF−2	0.057 5	0.073 0	0.094 1	0.124 9	0.132 5	0.136 3	0.137 9	0.134 1	0.133 3	0.128 2	0.125 9	0.123 3
CF−3	0.058 2	0.074 5	0.095 2	0.127 4	0.133 3	0.137 1	0.139 4	0.135 7	0.133 9	0.129 8	0.127 1	0.125 2
CF−4	0.060 1	0.079 2	0.101 4	0.132 3	0.141 7	0.144 6	0.145 7	0.143 8	0.140 9	0.136 5	0.132 2	0.129 3
CFAU	0.062 4	0.081 5	0.104 3	0.136 1	0.144 8	0.148 1	0.148 7	0.145 5	0.143 6	0.139 2	0.135 9	0.132 6

表9 D_3中不同n值下算法F_1值

Tab. 9 F_1 scores of algorithm under different n values in D_3

算法名称	n											
	2	3	5	10	15	20	25	30	35	40	45	50
UCF	0.049 9	0.065 5	0.085 9	0.114 8	0.121 0	0.127 3	0.129 1	0.126 5	0.121 2	0.119 3	0.117 2	0.115 4
CF−1	0.051 1	0.066 1	0.084 8	0.115 8	0.124 1	0.128 4	0.128 3	0.125 3	0.125 1	0.120 2	0.116 7	0.112 7
CF−2	0.054 8	0.069 2	0.088 6	0.121 4	0.129 5	0.132 5	0.134 2	0.129 9	0.130 6	0.124 1	0.122 9	0.119 8
CF−3	0.055 2	0.070 6	0.089 5	0.124 8	0.130 1	0.132 8	0.135 6	0.131 3	0.130 9	0.125 6	0.124 0	0.121 4
CF−4	0.057 4	0.074 6	0.095 5	0.128 6	0.139 3	0.141 3	0.142 4	0.140 6	0.137 9	0.132 0	0.128 3	0.124 3
CFAU	0.058 7	0.076 6	0.097 9	0.132 6	0.141 1	0.144 3	0.145 3	0.141 5	0.139 8	0.134 6	0.131 8	0.128 1

表 10　D_4 中不同 n 值下算法 F_1 值

Tab. 10　F_1 scores of algorithm under different n values in D_4

算法名称	n											
	2	3	5	10	15	20	25	30	35	40	45	50
UCF	0.049 1	0.064 4	0.084 6	0.113 3	0.119 3	0.125 6	0.127 5	0.125 7	0.119 5	0.117 7	0.115 0	0.114 0
CF−1	0.050 3	0.065 1	0.083 4	0.114 2	0.122 5	0.126 8	0.126 6	0.123 6	0.123 5	0.118 6	0.115 2	0.114 2
CF−2	0.054 0	0.068 1	0.087 2	0.119 8	0.127 9	0.130 8	0.132 5	0.128 2	0.129 0	0.122 4	0.121 4	0.118 3
CF−3	0.054 3	0.069 5	0.088 0	0.123 3	0.128 5	0.131 0	0.133 9	0.129 5	0.129 3	0.123 9	0.122 5	0.119 8
CF−4	0.058 8	0.073 2	0.094 0	0.126 9	0.137 7	0.139 6	0.140 6	0.138 9	0.136 2	0.130 2	0.126 6	0.126 6
CFAU	0.057 1	0.074 9	0.095 5	0.130 9	0.139 3	0.142 5	0.143 5	0.141 7	0.13 74	0.132 8	0.13 01	0.126 4

表 11　不同 n 值下算法 F_1 均值对比

Tab. 11　Comparison of algorithm F_1 means under different n values

算法名称	D_1	D_2	D_3	D_4
UCF	0.119 0	0.111 5	0.107 8	0.106 3
CF−1	0.119 8	0.112 1	0.108 2	0.107 0
CF−2	0.123 9	0.116 8	0.113 1	0.111 6
CF−3	0.125 5	0.118 1	0.114 3	0.122 8
CF−4	0.131 5	0.124 0	0.120 2	0.118 5
CFAU	0.135 3	0.126 9	0.122 7	0.121 0

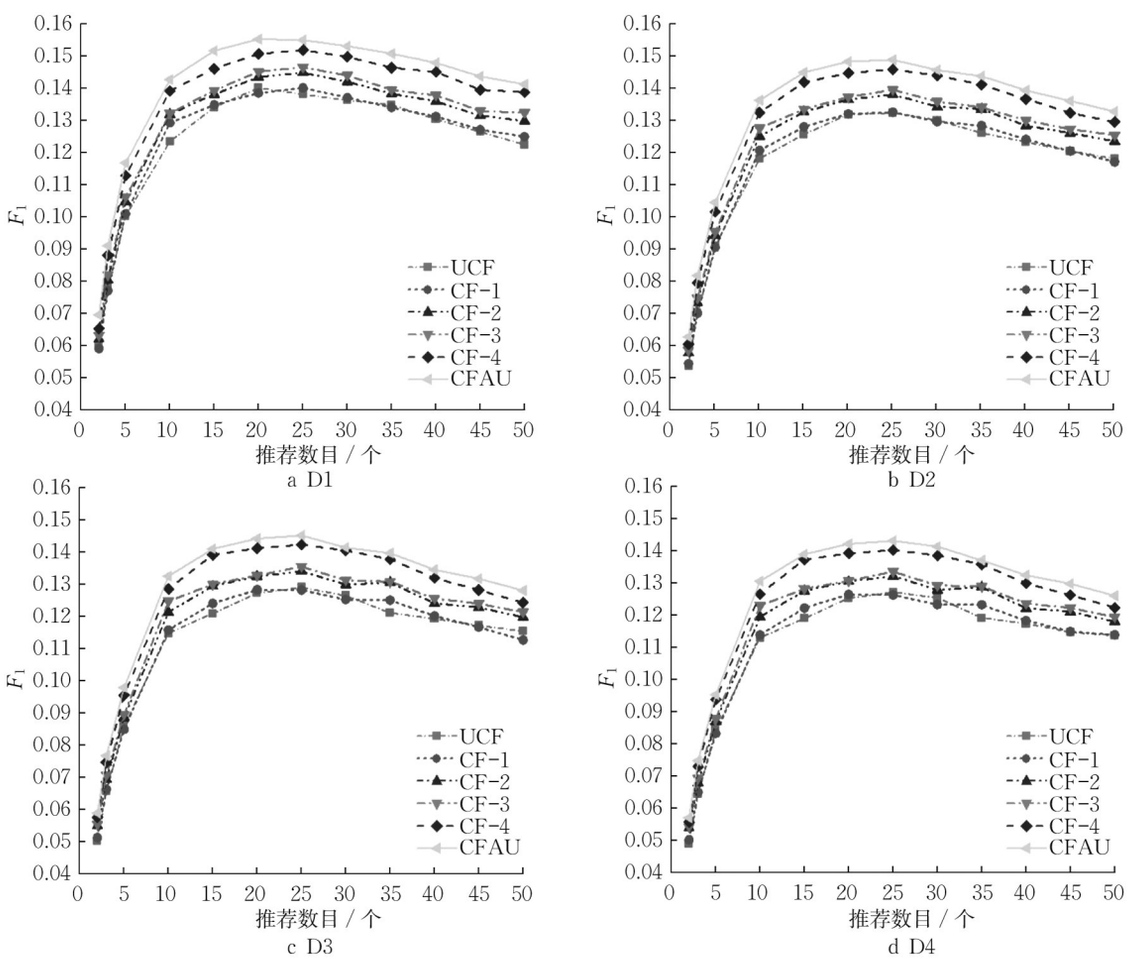

图 7　不同 n 值下的 F_1 对比曲线

Fig. 7　Comparison of F_1 under different n values

算问题,在CFAU算法计算复杂度比UCF算法高很多的前提下,仍能保证运行时间远小于UCF算法时间。在算法性能表现较好的MovieLens 1M数据集上,CFAU算法运行时间比UCF算法减少约

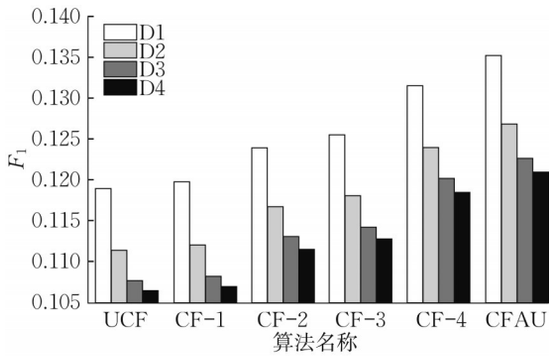

图 8 不同 n 值下的 F_1 均值对比

Fig. 8 Comparison of F_1 means under different n values

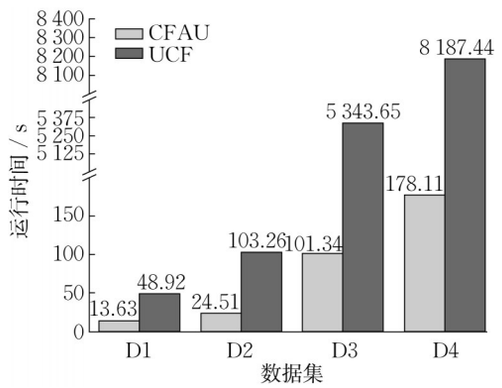

图 9 算法运行时间对比

Fig. 9 Comparison of algorithm running times

72.1%。

5 总结和展望

针对传统协同过滤算法存在的头部效应和推荐精度低问题,在基于用户的协同过滤算法基础上提出了一种基于自适应惩罚因子和用户评分行为的改进协同过滤算法。在皮尔逊相关系数的基础上引入了带有自适应参数的热门项目惩罚因子和与用户评分行为相关的用户评分偏好因子和用户时间因子,三者综合作用来加权用户相似度的计算。同时,倒排表的使用也大幅降低了改进算法的时间复杂度,让其可以在高计算复杂度的基础上保持较少的运行时间。在实验验证方面,选择了来自MovieLens和Amazon review data的4个数据集,并与传统算法UCF以及同领域内的4个改进算法进行了对比分析。实验结果表明,本文所提CFAU算法相较于对比算法有效提高了推荐质量,且具有一定的泛化性,同时较传统算法有效减少了运算时间。

在后续的研究中会考虑辅助信息的利用,如用户个人信息及项目属性信息,进一步提高算法个性化推荐的能力,同时也可以利用深度学习技术挖掘复杂的用户行为模式,进一步提高推荐系统的准确性和效率。

作者贡献声明:

赵晓群:指导论文写作,参与论文修改和定稿。

李煜堃:相关领域文献调研,收集数据并预处理,开展推荐性能对比实验与结果分析,论文撰写。

黄新林:提供论文课题与写作思路,对模型性能改进提出关键建议。

参考文献:

[1] 赵俊逸,庄福振,敖翔,等.协同过滤推荐系统综述[J].信息安全学报,2021,6(5):17.
ZHAO Junyi, ZHUANG Fuzhen, AO Xiang, *et al*. A review of collaborative filtering recommendation systems [J]. Journal of Information Security, 2021, 6(5): 17.

[2] 程华松,熊才权,柯远志,等.基于知识图谱的新闻推荐神经网络模型[J].湖北工业大学学报,2023,38(4):82.
CHENG Huasong, XIONG Caiquan, KE Yuanzhi, *et al*. A neural network model for news recommendation based on knowledge graphs [J]. Journal of Hubei University of Technology, 2023, 38(4): 82.

[3] 华泽,叶雨航.基于用户行为轨迹的在线音乐偏好模型[J].复旦学报(自然科学版),2022,61(3):342.
HUA Ze, YE Yuhang. An online music preference model based on user behavior trajectories [J]. Journal of Fudan University (Natural Science), 2022, 61(3): 342.

[4] 刘超慧,韩传福,陈天成,等.融合惩罚因子和时间权重的协同过滤推荐算法[J].信息技术与网络安全,2020,39(5):17.
LIU Chaohui, HAN Chuanfu, CHEN Tiancheng, *et al*. Collaborative filtering recommendation algorithm incorporating penalty factor and time weight [J]. Information Technology & Network Security, 2020, 39(5): 17.

[5] 周迅,刘超慧,周克萍,等.融入惩罚因子的协同过滤图书推荐算法[J].江苏科技信息,2022,39(17):76.
ZHOU Xun, LIU Chaohui, ZHOU Keping, *et al*. Collaborative filtering book recommendation algorithm with penalty factor integration [J]. Jiangsu Science and Technology Information, 2022, 39(17): 76.

[6] 马汉达,梁文德.基于左归一化图卷积网络的推荐模型[J].计算机应用,2023,43(202):111.
MA Handa, LIANG Wende. Recommendation model based on left-normalized graph convolutional network [J]. Computer Applications, 2023, 43(202): 111.

[7] 邹洋,吴和成,赵应丁,等.基于多权重相似度的随机漫步推荐算法[J].计算机应用研究,2020,37(11):3267.
ZOU Yang, WU Hecheng, ZHAO Yingding, *et al*. Random walk recommendation algorithm based on multi-weight

similarity [J]. Application Research of Computers, 2020, 37(11): 3267.

[8] 张卫国,袁炜轩,周熙然.融合深度去噪自编码器和注意力机制的推荐算法[J].计算机应用与软件,2023,40(8): 283.
ZHANG Weiguo, YUAN Weixuan, ZHOU Xiran. Recommendation algorithm integrating deep denoising autoencoder and attention mechanism [J]. Computer Applications and Software, 2023, 40(8): 283.

[9] 段琼,虞慧群,范贵生.融合商品流行度与信任度的混合推荐算法[J].华东理工大学学报(自然科学版),2022,48(4): 549.
DUAN Qiong, YU Huiqun, FAN Guisheng. Hybrid recommendation algorithm integrating product popularity and trustworthiness [J]. Journal of East China University of Science and Technology, 2022, 48(4): 549.

[10] 张洁,李港.一种基于评分信息熵的融合协同过滤算法[J].南京邮电大学学报(自然科学版),2021,41(2): 71.
ZHANG Jie, LI Gang. A collaborative filtering algorithm integrating rating information entropy [J]. Journal of Nanjing University of Posts and Telecommunications (Natural Science Edition), 2021, 41(2): 71.

[11] SHUAI Jie, ZHANG Kun, Wu Le, *et al*. A review-aware graph contrastive learning framework for recommendation [C]//Proceedings of the 45th International ACM SI GIR Conferenceon Researchand Development in Information Retrieval. New York: Association for Computing Machinery, 2022:1283.

[12] YU J, YIN H, LI J, *et al*. Self-supervised multi-channel hypergraph convolutional network for social recommendation [C]//Proceedings of the Web Conference 2021. NewYork: Association for Computing Machinery, 2021: 413-424.

[13] YU Q, ZHAO M, LUO Y. Collaborative Filtering Hybrid Recommendation Algorithm based on Optimal Weight [C]// 2022 International Conference on Computer Engineering and Artificial Intelligence (ICCEAI). [S. l.]: IEEE, 2022: 144-147.

[14] WANG Z, CHEN H, LI Z, *et al*. VRConvMF: visual recurrent convolutional matrix factorization for movie recommendation [J]. IEEE Transactions on Emerging Topics in Computational Intelligence, 2022, 6(3): 519.

[15] TRAN N T, LAUW H W. Aligning dual disentangled user representations from ratings and textual content [C]// Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. [S.l.]: [s.n.], 2022: 1798-1806..

[16] ZHOU X, SHEN Z. A tale of two graphs: Freezing and denoising graph structures for multimodal recommendation [C]//Proceedings of the 31st ACM international conference on multimedia. [S.l.]: [s.n.],2023: 935-943.

[17] ZOU F, CHEN D, XU Q, *et al*. A two-stage personalized recommendation based on multi-objective teaching - learning-based optimization with decomposition [J]. Neurocomputing, 2021, 452: 716.

[18] XU S, ZHUANG H, SUN F, *et al*. Recommendation algorithm of probabilistic matrix factorization based on directed trust [J]. Computers & Electrical Engineering, 2021, 93: 107206.

[19] WENG L J, ZHANG Q S, LIN Z B, *et al*. Harnessing heterogeneous social networks for better recommendations: a grey relational analysis approach [J]. Expert Systems with Applications, 2021, 174: 1.

[20] SALAMAT A, LUO X, JAFARI A. HeteroGraphRec: a heterogeneous graph-based neural networks for social recommendations [J]. Knowledge-Based Systems, 2021, 217: 106817.

[21] 杨北辰,余粟,陈乐.一种2阶段联合预测的协同过滤推荐算法[J].信息安全研究,2023,9(3): 291.
YANG Beichen, YU Su, CHEN Le. A collaborative filtering recommendation algorithm with two-stage joint prediction [J]. Journal of Information Security Research, 2023, 9(3): 291.

[22] 张婷婷,欧阳丹彤,孙成林,等.融合协同过滤的神经Bandits推荐算法[J].吉林大学学报(理学版),2024,62(1): 92.
ZHANG Tingting, OUYANG Dantong, SUN Chenglin, *et al*. Neural Bandits recommendation algorithm integrating collaborative filtering [J]. Journal of Jilin University (Science Edition), 2024, 62(1): 92.

[23] 黄晓宇,鲜征征,杨雄文,等.基于"少数派报告"的协同过滤模型[J].中国管理科学,2024,32(3): 20.
HUANG Xiaoyu, XIAN Zhengzheng, YANG Xiongwen, *et al*. Collaborative filtering model based on "minority report" [J]. Chinese Journal of Management Science, 2024, 32(3): 20.

[24] 李驰,游小钰,张谧.基于解耦图卷积网络的协同过滤推荐模型[J].中文信息学报,2023,37(11): 131.
LI Chi, YOU Xiaoyu, ZHANG Mi. Collaborative filtering recommendation model based on decoupled graph convolutional network [J]. Journal of Chinese Information Proceedingsessing, 2023, 37(11): 131.

[25] 姬璐,于万钧,陈颖.融合兴趣差异和评分差异的协同过滤推荐算法[J].计算机工程与设计,2023,44(6): 1721.
JI Lu, YU Wanjun, CHEN Ying. Collaborative filtering recommendation algorithm integrating interest and rating differences [J]. Computer Engineering and Design, 2023, 44(6): 1721.

[26] 朱金侠,孟祥福,邢长征,等.融合图卷积注意力机制的协同过滤推荐方法[J].智能系统学报,2023,18(6): 1295.
ZHU Jinxia, MENG Xiangfu, XING Changzheng, *et al*. Collaborative filtering recommendation method integrating graph convolutional attention mechanism [J]. CAAI Transactions on Intelligent Systems, 2023, 18(6): 1295.