DOI：10.13718/j.cnki.xdzk.2021.10.003

多通道特征向量的新三角距离高效推荐

吕亚兰，　张恒汝，　秦　琴，　徐媛媛

西南石油大学 计算机科学学院，成都 610500

摘要：为同时提高推荐系统的准确度和效率，提出了一种多通道特征向量的新三角距离推荐算法. 首先从原始评分矩阵中提取多通道特征向量；其次结合三角距离和 Jaccard 系数构建新三角距离；最后将该距离用于 k 近邻算法以表征两个项目间的相似度. 在 4 个真实数据集上的实验结果表明：文中提出的算法推荐效率更高，并能保持较好的推荐准确度.

关　键　词：多通道特征向量；新三角距离；高效推荐；k 近邻算法

中图分类号：TP391　　**文献标志码**：A

文 章 编 号：1673－9868(2021)10－0019－10

开放科学(资源服务)标识码(OSID)：

Efficient Recommendation of New Triangular Distance for Multi-channel Feature Vectors

LYU Yalan，　ZHANG Hengru，　QIN Qin，　XU Yuanyuan

College of Computer Science，Southwest Petroleum University，Chengdu 610500，China

Abstract：In order to simultaneously enhance the accuracy and efficiency of the recommender system，this paper designs a new triangular distance recommendation algorithm for multi-channel feature vectors. Firstly，we use item's rating matrix to extract multi-channel feature vectors. Secondly，we combine the triangular distance and Jaccard similarity coefficient to conduct a new triangular distance. Finally，we apply this distance to the k-nearest neighbor algorithm to characterize the similarity between two items. The experimental results on 4 real datasets show that the proposed algorithm is more efficient and better accuracy.

Key words：multi-channel feature vector；new triangular distance；efficient recommendation；k-nearest neighbor algorithm

推荐系统是目前解决信息过载的有效手段. 协同过滤[1]是主流的推荐算法之一，它利用历史评分数据来获取用户对项目的偏好. 协同过滤按照不同的实现方式可以分为基于 k 近邻[2]、基于矩阵分解[3]以及基于神经网络的协同过滤算法[4]等. k 近邻利用历史评分获取 k 个具有相似偏好的用户或者具有相似属性的项目[2]，常用表征用户或项目相似度的距离有：Cosine[5]，PCC(pearson correlation coefficient)[6]，Jac-

收稿日期：2021－05－25

基金项目：国家自然科学基金项目(61902328)；四川省科技厅应用基础研究项目(2019YJ0314)；四川省青年科技创新研究团队(2019JDTD0017)；浙江海洋大学大数据挖掘与应用省重点实验室开放课题(OBDMA202005).

作者简介：吕亚兰，硕士研究生，主要从事推荐系统的研究.

通信作者：张恒汝，教授.

card[7]和CPC(constrained pearson correlation)[8]. 然而这些算法大都采用用户或者项目的全局评分来计算相似度，导致其时间复杂度较高，推荐效率低.

本文提出了一种多通道特征向量的新三角距离推荐算法(new triangular distance recommendation algorithm for multi-channel feature vector，NTRFC). 算法的输入为从原始评分矩阵中提取的多通道特征向量(简称特征向量)，在 k 近邻算法中采用新三角距离，从而提高推荐效率并保持较好的推荐准确度.

首先，从原始评分矩阵中提取得到特征向量，其通道数目为原始评分矩阵中评分等级的数目[9]，将其作为输入，可有效降低算法的复杂度. 假定评分矩阵有 n 个用户，m 个项目，以及 l 个评分等级. 以原始评分矩阵为输入，计算相似度的时间复杂度为 $O(nm)$，而以多通道特征向量为输入，计算相似度的时间复杂度是 $O(lm)$. 评分矩阵中用户数目 n 远远大于评分等级数目 l，故 $O(lm)$远远小于 $O(nm)$. 例如，数据集Amazon(http://snap.stanford.edu/data/web-Amazonlinks.html)和Movielens943u (https://grouplens.org/datasets/movielens/100k/)的评分等级均为1～5分，故它们的通道数目为5，即每个项目的特征向量长度为5.

其次，利用两个项目的特征向量构建新三角距离. 该距离将三角距离和Jaccard系数结合. 这是因为在提取特征向量后，损失了用户、项目以及评分之间的关系信息，仅保留用户对项目评分的数量信息. 若仅考虑三角距离，则无法精确判断项目之间的相似度. 考虑到Jaccard系数能充分利用共同评分项目数占所有项目数的比值信息，故结合Jaccard系数，从而在一定程度上弥补了原始评分信息.

最后，将设计的新三角距离用于 k 近邻算法中，以判断两个项目的相似度. 本文提出的NTRFC算法与基于其他距离的 k 近邻算法在4个真实数据集上进行对比实验，利用6种准确度指标和运行时间进行评价. 实验结果表明：NTRFC算法运行时间低于已有算法，并在大部分准确度指标上占优.

1 相关工作

本节介绍评分系统[10]定义和常见的几种距离，本文使用的符号见表1.

表1 符号系统

符号	含义
U	所有用户的集合
T	所有项目的集合
u_i	第 i 个用户
t_p	第 p 个项目
$\boldsymbol{v}_p$	项目 t_p 的多通道特征向量
I_p	对项目 t_p 评过分的用户集合
C	所有评分等级构成的集合
l	通道数 l
$\boldsymbol{R}$	评分矩阵
H_p	项目 t_p 的邻居集合
$G(t_p, t_q)$	对第 p,q 个项目均评过分的用户集合
$P(t_p, x)$	项目 t_p 的评分为 x 的概率
r_{ip}	第 i 个用户对第 p 个项目的实际评分
r_{ip}^*	第 i 个用户对第 p 个项目的预测评分
$\overrightarrow{r(\cdot, t_p)}$	项目 t_p 的评分向量
$\overline{r(\cdot, t_p)}$	项目 t_p 的平均评分

1.1 评分系统

现回顾评分系统[10]的定义，令 $U=\{u_1, u_2, \cdots, u_n\}$ 为一个推荐系统的用户集合，令 $T=\{t_1, t_2, \cdots, t_m\}$ 为推荐给用户的项目集合，由此，评分函数定义为

$$\boldsymbol{R}: U \times T \rightarrow C \tag{1}$$

其中，$\boldsymbol{R}$ 为一个 $n \times m$ 的评分矩阵；$\boldsymbol{R}=(r_{ip})_{n\times m}$；$C$ 表示用户评价每个项目的评分等级构成的集合，如 $C=\{1, 2, 3, 4, 5\}$.

表 2 给出了一个用户数为 5 和项目数为 6 的评分矩阵. 评级为 1～5 分，则通道数为 5. 评分反映出用户对项目的喜爱程度，分值越高表示用户越喜爱该项目，0 表示用户未给项目评分. r_{ip} 表示用户 u_i 给项目 t_p 的实际评分，$G(t_p, t_q)$ 表示对项目 t_p 和 t_q 共同评分的用户集合. 例如，$r_{12}=3$ 表示用户 u_1 给项目 t_2 评分为 3 分，$G(t_1, t_2)=\{u_1, u_4\}$ 表示对项目 t_1 和 t_2 共同评分的用户是 u_1 和 u_4.

表 2　评分矩阵($\boldsymbol{R}$)

用户\项目	t_1	t_2	t_3	t_4	t_5	t_6
u_1	2	3	2	2	4	0
u_2	2	0	4	1	5	0
u_3	0	0	4	5	0	0
u_4	4	2	5	4	0	5
u_5	0	4	1	2	0	4

1.2　已有的距离

k 近邻算法通常计算用户或项目之间的距离来寻找用户或项目的邻居，从而预测用户对项目的评分. 表 3 列出了 9 个常用距离度量公式，并分析它们的时间复杂度.

表 3 中，Cosine[5]，ED[11]，BC[12]，PCC[6]，MD[13]，SØrensen[14-15]，Canberra[16]，Lorentzian[17] 和 Divergence[18] 距离的时间复杂度均为 $O(n)$，但 BC[12] 距离的时间复杂度为 $O(l)$. 其中，n 表示输入向量的长度，l 表示评分的等级数.

表 3　不同距离公式

距离	公　　式	时间复杂度
Cosine[5]	$\text{Cosine}(t_p, t_q)=\dfrac{\overrightarrow{r(\cdot, t_p)}\cdot\overrightarrow{r(\cdot, t_q)}}{\lvert\overrightarrow{r(\cdot, t_p)}\rvert\times\lvert\overrightarrow{r(\cdot, t_q)}\rvert}$	$O(n)$
ED[11]	$\text{ED}(t_p, t_q)=\sqrt{\sum_{u_i\in G_{t_p, t_q}}(r_{ip}-r_{iq})^2}$	$O(n)$
BC[12]	$\text{BC}(t_p, t_q)=\sum_{x\in C}\sqrt{P(t_p, x)\times P(t_q, x)}$	$O(l)$
PCC[6]	$\text{PCC}(t_p, t_q)=\dfrac{\sum_{u_i\in G_{t_p, t_q}}(r_{ip}-\overline{r(\cdot, t_p)})(r_{iq}-\overline{r(\cdot, t_q)})}{\sqrt{\sum_{u_i\in G_{t_p, t_q}}(r_{ip}-\overline{r(\cdot, t_p)})^2}\sqrt{\sum_{u_i\in G_{t_p, t_q}}(r_{iq}-\overline{r(\cdot, t_q)})^2}}$	$O(n)$
MD[13]	$\text{MD}(t_p, t_q)=\sum\lvert\overrightarrow{r(\cdot, t_p)}-\overrightarrow{r(\cdot, t_q)}\rvert$	$O(n)$
SØrensen[14-15]	$\text{SØrensen}(t_p, t_q)=\dfrac{\sum\lvert\overrightarrow{r(\cdot, t_p)}-\overrightarrow{r(\cdot, t_q)}\rvert}{\sum\lvert\overrightarrow{r(\cdot, t_p)}+\overrightarrow{r(\cdot, t_q)}\rvert}$	$O(n)$
Canberra[16]	$\text{Canberra}(t_p, t_q)=\sum\dfrac{\lvert\overrightarrow{r(\cdot, t_p)}-\overrightarrow{r(\cdot, t_q)}\rvert}{\overrightarrow{r(\cdot, t_p)}+\overrightarrow{r(\cdot, t_q)}}$	$O(n)$
Lorentzian[17]	$\text{Lorentzian}(t_p, t_q)=\sum\ln(1+\lvert\overrightarrow{r(\cdot, t_p)}-\overrightarrow{r(\cdot, t_q)}\rvert)$	$O(n)$
Divergence[18]	$\text{Divergence}(t_p, t_q)=2\sum\dfrac{(\overrightarrow{r(\cdot, t_p)}-\overrightarrow{r(\cdot, t_q)})^2}{(\overrightarrow{r(\cdot, t_p)}+\overrightarrow{r(\cdot, t_q)})^2}$	$O(n)$

2 NTRFC

NTRFC首先利用原始评分矩阵提取特征向量，然后基于特征向量设计新三角距离，最后将新三角距离应用到 k 近邻算法中.

2.1 特征向量的提取

项目的评分等级构成通道集合 C. 例如，当评分等级为1～5分时，通道集合 $C=\{1, 2, 3, 4, 5\}$. 该集合包含有通道1～5，通道数 l 为5. 为了处理项目的离散评分，本文将每个项目的评分映射到多个通道.

用户 u_i 对项目 t_p 的评分 r_{ip} 与通道的关系为

$$a(u_i, t_p, c)=\begin{cases}1 & \text{if } r_{ip}=c\\ 0 & \text{otherwise}\end{cases} \tag{2}$$

其中 c 表示当前通道数值.

当 r_{ip} 与 c 相等时，连接用户 u_i 和通道 c 的边的数量加1. 项目 t_p 上通道 c 的连接数为

$$d(t_p, c)=\sum_{i=1}^{n}a(u_i, t_p, c) \tag{3}$$

对于长度为 l 的通道，项目 t_p 提取后的特征向量为

$$\boldsymbol{v}_p=[d(t_p, c_1), d(t_p, c_2), \cdots, d(t_p, c_l)] \tag{4}$$

以表2展示的评分矩阵为例，项目 t_1 对应的特征向量为 $\boldsymbol{v}_1=[0, 2, 0, 1, 0]$，如图1.

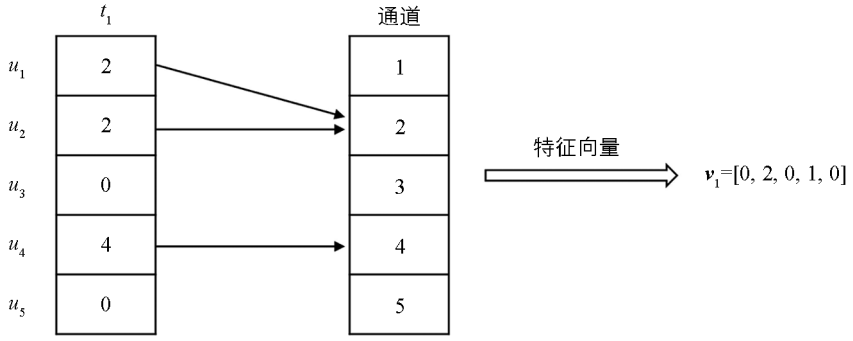

图1 多通道特征向量的构建

2.2 新三角距离

利用特征向量，设计新三角距离公式为

$$\mathrm{NTJ}(\boldsymbol{v}_p, \boldsymbol{v}_q, t_p, t_q)=NT(\boldsymbol{v}_p, \boldsymbol{v}_q)\times\mathrm{Jaccard}(t_p, t_q) \tag{5}$$

其中，$\boldsymbol{v}_p$,$\boldsymbol{v}_q$ 分别为项目 t_p,t_q 的特征向量. $NT(\boldsymbol{v}_p, \boldsymbol{v}_q)$为三角距离，$\mathrm{Jaccard}(t_p, t_q)$为Jaccard系数.

$NT(\boldsymbol{v}_p, \boldsymbol{v}_q)$为

$$NT(\boldsymbol{v}_p, \boldsymbol{v}_q)=1-\frac{2\times\|\boldsymbol{v}_p-\boldsymbol{v}_q\|}{\|\boldsymbol{v}_p\|+\|\boldsymbol{v}_q\|} \tag{6}$$

其中 $\|\cdot\|$ 为向量的二范数. $\mathrm{Jaccard}(t_p, t_q)$为

$$\mathrm{Jaccard}(t_p, t_q)=\frac{|I_p\cap I_q|}{|I_p\cup I_q|} \tag{7}$$

其中，I_p 为对项目 t_p 评过分的用户集合；I_q 为对项目 t_q 评过分的用户集合；$|\cdot|$表示集合的基.

为了更准确地描述项目之间的相似度，新三角距离引入Jaccard系数. 这是由于原始评分矩阵进行特征向量提取后，损失了用户、项目以及评分之间的对应关系信息，只保留了用户对项目评分的数量信息. 如果仅使用三角距离或其他一般距离则无法准确计算项目之间的相似度. 以表2中项目 t_5 和 t_6 为例，通

过提取后它们的特征向量 $\boldsymbol{v}_5$ 和 $\boldsymbol{v}_6$ 均为[0，0，0，1，1]，使用三角距离计算后相似度为 1，使用 Cosine 距离计算后相似度也为 1. 但实际上，t_5 和 t_6 的评分分别来源于完全不同的用户 u_1，u_2 和 u_4，u_5. 使用新三角距离计算得到相似度为 0，更加合理.

以表 2 展示的评分矩阵为例，使用新三角距离计算项目 t_1 和 t_2 相似度流程如下：

1) 提取项目 t_1 和 t_2 的特征向量 $\boldsymbol{v}_1=[0，2，0，1，0]$和 $\boldsymbol{v}_2=[0，1，1，1，0]$.

2) 计算 NT 距离为

$$NT=1-\frac{2\times\sqrt{(0-0)^2+(2-1)^2+(0-1)^2+(1-1)^2+(0-0)^2}}{\sqrt{2^2+1^2}+\sqrt{1^2+1^2+1^2}}\approx 0.29$$

计算 Jaccard 系数为 $\frac{|I_1\cap I_2|}{|I_1\cup I_2|}=\frac{2}{5}=0.40$，故 $NTJ(\boldsymbol{v}_1，\boldsymbol{v}_2，t_1，t_2)=\mathrm{NT}*\mathrm{Jaccard}=0.29*0.4=0.116$.

2.3　基于新三角距离的 k 近邻算法

将新三角距离应用到 k 近邻算法[19]中，预测用户对项目的评分. 其计算公式[20]定义为

$$r_{ip}^{*}=\overline{r(\cdot，t_p)}+\frac{\sum\limits_{t_q\in H_p}NTJ(\boldsymbol{v}_p，\boldsymbol{v}_q，t_p，t_q)(r_{iq}-\overline{r(\cdot，t_q)})}{\sum\limits_{t_q\in H_p}\mathrm{NTJ}(\boldsymbol{v}_p，\boldsymbol{v}_q，t_p，t_q)}\tag{8}$$

其中，r_{ip}^{*} 表示用户 u_i 给物品 t_p 的预测评分；$\overline{r(\cdot，t_p)}$和 $\overline{r(\cdot，t_q)}$分别为项目 t_p 和 t_q 的平均评分；r_{iq} 为用户 u_i 给物品 t_q 的实际评分；H_p 表示项目 t_p 的邻居集合.

以表 2 为例，使用基于新三角距离的 k 近邻算法预测用户 u_3 对 t_1 的评分 r_{31}^{*} 流程如下：

1) 分别提取项目 t_1，t_3 和 t_4 的特征向量 $\boldsymbol{v}_1=[0，2，0，1，0]$，$\boldsymbol{v}_3=[1，1，0，2，1]$和 $\boldsymbol{v}_4=[1，2，0，1，1]$.

2) 使用新三角距离分别计算项目 t_1 和 t_3，t_4 之间的相似度 $\mathrm{NTJ}(\boldsymbol{v}_1，\boldsymbol{v}_3，t_1，t_3)=0.11$，$\mathrm{NTJ}(\boldsymbol{v}_1，\boldsymbol{v}_4，t_1，t_4)=0.25$.

3) 用户 u_3 对 t_1 的预测评分 $r_{31}^{*}=1.6+\frac{0.11\times(4-1.6)+0.25\times(5-1.6)}{0.11+0.25}\approx 4.69$.

2.4　算法描述

算法总结了 NTRFC 的具体步骤. 步骤 1 读取并初始化评分数据；步骤 2 根据式(2)至式(4)为每一个项目提取多通道特征向量；步骤 3 初始化 k 个邻居，并计算与邻居的新三角距离，得到最远距离 D；步骤 4 至步骤 10 根据式(5)至式(7)计算与其余项目之间的新三角距离，并得到最终 k 个最近邻居；步骤 11 根据式(8)计算预测评分.

算法　NTRFC

输入：用户-项目评分矩阵 $\boldsymbol{R}$

输出：预测评分 r_{ip}^{*}

step 1：初始化评分数据

step 2：根据式(2)至式(4)提取特征向量

step 3：初始化 k 个邻居，并计算新三角距离，得到最远距离 D

step 4：for 其余有评分的项目 do

step 5：根据式(5)至式(7)计算新三角距离 d

step 6：if ($d<D$)

step 7：用该项目替代最远距离项目

step 8：$D=d$

step 9：end if

step10：end for

step11：根据式(8)得到预测评分 r_{ip}^{*}

算法时间复杂度分析如下：步骤1读取评分数据的时间为 $O(nm)$；步骤2提取多通道特征向量的时间为 $O(nm)$；步骤3初始化并计算与 k 个邻居的距离时间为 $O(kl)$；步骤4至步骤10获得最近 k 个邻居的时间为 $O(ml)$；步骤11预测评分的时间为 $O(k)$. 故整个模型的时间复杂度为 $O(nm)$.

3 实 验

针对提出的算法进行两组对比实验，用来回答以下两个问题：1) 本文算法是否能提高推荐效率？2) 本文提出的新三角距离能否保证较好的推荐准确度？

问题一中采用特征向量或原始评分矩阵作为输入，使用本文提出的NTJ距离与另外9种距离计算项目间的相似度，利用 k 近邻算法进行协同过滤推荐，比较两者的运行时间从而判断何种输入下的推荐效率更高.

问题二比较使用NTJ距离或其他9种距离的 k 近邻算法推荐准确度的高低.

3.1 数据集

本文使用Amazon，Movielens943u，Movielens706u（https://grouplens.org/datasets/movielens/100k/)和Eachmovie（http://www.research.digital.com/SRC/eachmovie/)数据集. 表4给出了它们的基本信息，前3个数据集采用的评分等级是1～5分，Eachmovie数据集采用的评分等级是0.2～1分，0分表示用户没有给项目评分. 在提取特征向量时，将Eachmovie数据集的评分等级扩展为1～5分，预测后按比例还原.

表4 数据集

数据集	用户个数	项目个数	评分个数
Amazon	1 094	1 673	34 120
MovieLens943u	943	1 684	100 000
MovieLens706u	706	8 570	100 023
Eachmovie	72 916	1 628	2 811 983

3.2 实验设计

通过两组实验Exp1和Exp2分别回答本节开始提出的两个问题.

Exp1：比较输入分别为特征向量和原始评分矩阵的算法的运行时间. 使用本文提出的NTJ距离与另外9种距离计算项目间的相似度，并将其应用于 k 近邻算法. 记录不同输入下，使用同样距离公式的算法在4个数据集下的运行时间，运行时间越少，表示推荐效率越高.

Exp2：在输入为特征向量时，分别采用本文提出的NTJ距离与另外9种距离进行推荐准确度对比实验.

在本文使用的 k 近邻算法中，设置两个参数 LR 和 TR. LR 表示用户是否喜欢某项目的门限值，设置为3. TR 表示是否给用户推荐某项目的门限值，设置为3.5.

采用交叉验证的方式进行实验，首先将原始评分随机分为5等份，从中选取其中4份作为训练集，1份作为测试集；其次，提取多通道特征向量，结合不同的距离预测评分；最后，通过6个指标来衡量预测评分与真实评分的差距. 上述步骤重复5次，每个指标下将得到5次不同的数据，将这些指

标平均后做对比实验.

表 5 给出了 6 个准确度评价指标.

表 5　评价指标

指标名称	公　　式
MAE[21] ↓	$\mathrm{MAE}=\frac{\sum_{i=0}^{n-1}\sum_{p=0}^{m-1}\lvert r_{ip}-r_{ip}^{*}\rvert}{\lvert(i,\ p)\in\{0..n-1\}\times\{0..m-1\}\lvert r_{ip}\neq 0\rvert}$
RMSE[21] ↓	$\mathrm{RMSE}=\sqrt{\frac{\sum_{i=0}^{n-1}\sum_{p=0}^{m-1}(r_{ip}-r_{ip}^{*})^{2}}{\lvert(i,\ p)\in\{0..n-1\}\times\{0..m-1\}\lvert r_{ip}\neq 0\rvert}}$
Accuracy[22] ↑	$\mathrm{Acc}=\frac{\lvert\{\langle u_i,\ t_p\rangle\lvert r_{ip}>LR,\ r_{ip}^{*}>TR\}\rvert+\lvert\{\langle u_i,\ t_p\rangle\lvert r_{ip}<LR,\ r_{ip}^{*}<TR\}\rvert}{\lvert r_{ip}^{*}\rvert}$
Recall[22] ↑	$\mathrm{Recall}=\frac{\lvert\{\langle u_i,\ t_p\rangle\lvert r_{ip}>LR,\ r_{ip}^{*}>TR\}\rvert}{\lvert\{\langle u_i,\ t_p\rangle\lvert r_{ip}>LR,\ r_{ip}^{*}>TR\}\rvert+\lvert\{\langle u_i,\ t_p\rangle\lvert r_{ip}>LR,\ r_{ip}^{*}<TR\}\rvert}$
Precision[22] ↑	$\mathrm{Pre}=\frac{\lvert\{\langle u_i,\ t_p\rangle\lvert r_{ip}>LR,\ r_{ip}^{*}>TR\}\rvert}{\lvert\{\langle u_i,\ t_p\rangle\lvert r_{ip}>LR,\ r_{ip}^{*}>TR\}\rvert+\lvert\{\langle u_i,\ t_p\rangle\lvert r_{ip}<LR,\ r_{ip}^{*}>TR\}\rvert}$
F1[22] ↑	$\mathrm{F1}=\frac{2*\mathrm{Precision}*\mathrm{Recall}}{\mathrm{Precision}+\mathrm{Recall}}$

注："↓"表示该指标"越小越好"，"↑"表示该指标"越大越好".

3.3　实验结果

3.3.1　Exp1 的结果

在 4 个数据集上，分别使用特征向量和原始评分作为输入，采用本文提出的 NTJ 距离和其余 9 种已有距离计算项目相似度的 k 近邻算法的运行时间结果，如图 2，其中图 2(d)在 Eachmovie 上的运行时间进行了对数处理.

以 NTJ 距离为例，对运行时间结果进行简要分析. 在 Amazon 数据集上，采用特征向量作为输入使得运行时间下降了 39.33%，如图 2(a). 在 Movielens943u 数据集上，采用特征向量作为输入使得运行时间下降了 48.79%，如图 2(b). 在 Movielens706u 数据集上，采用特征向量作为输入使得运行时间下降了 40.67%，如图 2(c). 在 Eachmovie 数据集上，采用特征向量作为输入使得运行时间下降了 52.54%，如图 2(d).

综上所述，使用特征向量作为输入，能有效提高算法效率并大幅减少运行时间.

3.3.2　Exp2 的结果

不同距离在 4 个数据集上的准确度结果如表 6. 每一个子表包括 5 次实验的平均结果、标准差和平均性能排名，NTRFC 为本文提出的算法，括号中的数字表示当前距离的性能排名. 若平均结果相同，则比较标准差，标准差越小，性能越好.

在 Amazon 数据集上，本文提出的 NTRFC 算法在 F1，Accuracy，Precision，MAE 及 RMSE 评价指标上排名第一，如表 6(a)所示. 在 Movielens943u 数据集上，本文提出的 NTRFC 算法在 F1，Accuracy，Precision 及 Recall 评价指标上排名第一，如表 6(b)所示. 在 Movielens706u 数据集上，本文提出的 NTRFC 算法在 F1，Precision 及 MAE 评价指标上排名第一，如表 6(c)所示. 在 Eachmovie 数据集上，本文提出的 NTRFC 算法在 F1，Accuracy 及 Recall 评价指标上排名第一，如表 6(d)所示. 总体而言，在 4 个数据集

上，NTRFC算法在个数准确度指标上占优.

综上所述，本文提出的NTRFC算法能有效提高算法效率节省时间，并保持较好的推荐准确度.

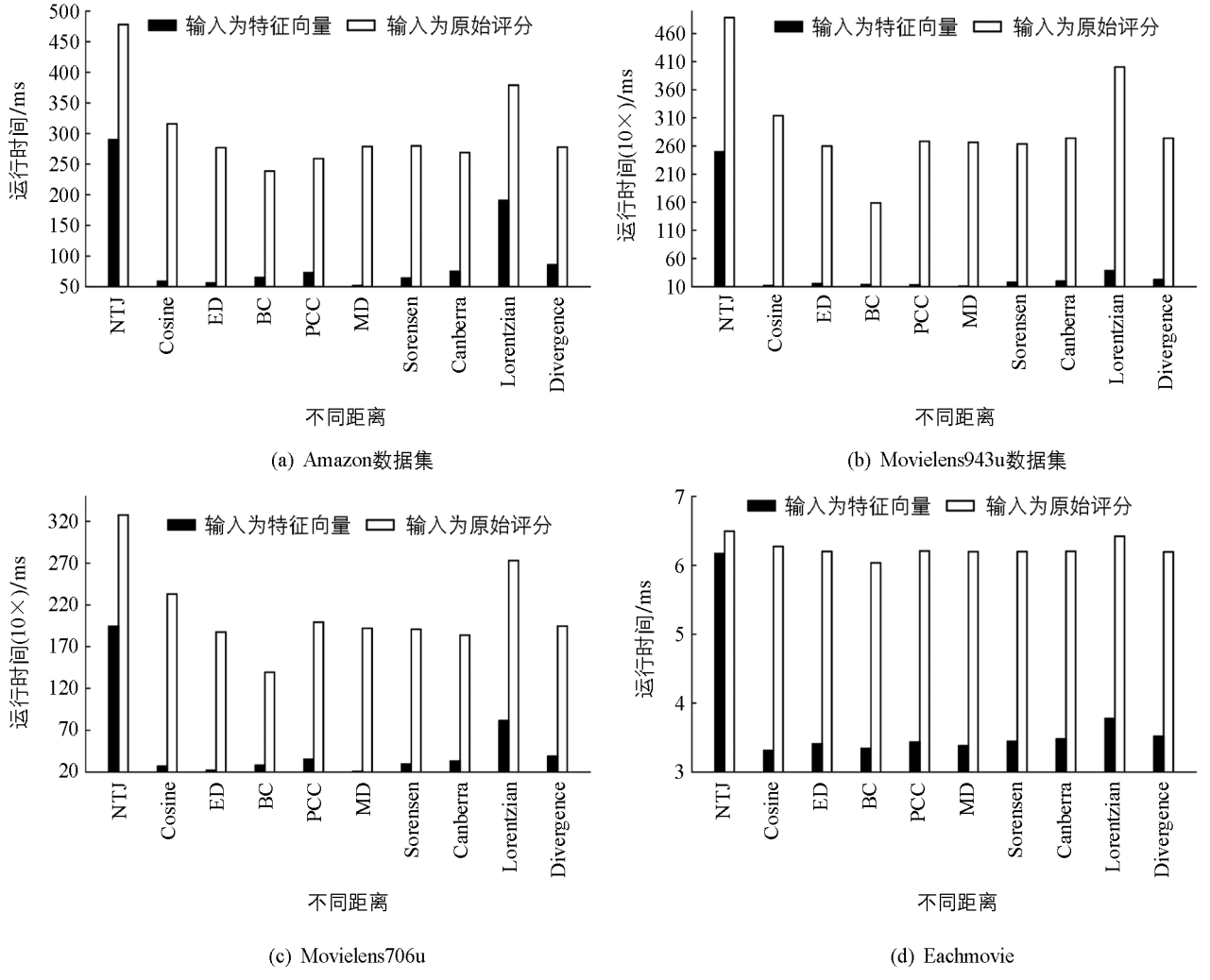

图2 4个数据集上的运行时间结果

表6 6个评价指标下的实验结果对比

(a) Amazon数据集. 在该数据集上，本文提出的算法在5个指标上占优.

	F1↑	Accuracy↑	Precision↑	MAE↓	Recall↑	RMSE↓
NTRFC	0.912 6±0.002 9(1)	0.867 4±0.003 0(1)	0.966 5±0.002 1(1)	0.236 9±0.005 2(1)	0.864 5±0.003 5(10)	0.514 8±0.011 4(1)
Cosine	0.895 1±0.001 9(2)	0.864 8±0.002 7(2)	0.915 7±0.002 2(3)	0.440 7±0.006 5(3)	0.875 5±0.002 8(6)	0.741 4±0.007 4(2)
ED	0.877 0±0.001 2(9)	0.846 8±0.003 3(10)	0.879 4±0.003 7(8)	0.643 6±0.006 4(10)	0.874 7±0.003 2(8)	0.934 4±0.011 2(10)
BC	0.893 1±0.002 0(4)	0.858 9±0.003 6(4)	0.916 8±0.003 1(2)	0.432 2±0.009 1(2)	0.870 6±0.003 2(9)	0.744 1±0.013 8(3)
PCC	0.893 2±0.002 0(3)	0.862 9±0.002 3(3)	0.912 4±0.002 8(4)	0.473 9±0.007 2(4)	0.874 8±0.001 9(7)	0.782 0±0.009 3(4)
MD	0.877 4±0.001 2(8)	0.847 6±0.002 4(9)	0.879 3±0.003 3(9)	0.640 4±0.005 8(9)	0.875 6±0.001 5(5)	0.932 5±0.010 6(9)
Sorensen	0.879 4±0.001 3(10)	0.851 1±0.003 2(7)	0.878 2±0.003 1(10)	0.638 0±0.007 0(8)	0.880 5±0.001 6(2)	0.928 1±0.012 1(8)
Canberra	0.880 5±0.001 1(6)	0.853 1±0.002 3(6)	0.881 9±0.003 1(6)	0.627 4±0.006 5(6)	0.879 1±0.001 6(3)	0.916 2±0.011 1(6)
Lorentzian	0.882 2±0.000 7(5)	0.855 6±0.001 8(5)	0.882 0±0.002 9(5)	0.620 4±0.005 3(5)	0.882 4±0.001 7(1)	0.908 3±0.009 8(5)
Divergence	0.879 3±0.001 3(7)	0.851 0±0.002 8(8)	0.880 7±0.003 3(7)	0.634 7±0.006 9(7)	0.877 9±0.001 6(4)	0.926 2±0.012 3(7)

续表 6

(b) Movielens943u 数据集. 在该数据集上，本文提出的算法在 4 个指标上占优.

	F1↑	Accuracy↑	Precision↑	MAE↓	Recall↑	RMSE↓
NTRFC	0.700 6±0.010 4(1)	0.662 9±0.012 4(1)	0.778 8±0.004 1(1)	0.740 6±0.007 5(3)	0.636 7±0.015 1(1)	0.956 1±0.010 9(4)
Cosine	0.697 6±0.013 4(3)	0.659 4±0.015 9(3)	0.775 6±0.006 9(4)	0.746 1±0.011 7(4)	0.633 9±0.017 9(2)	0.948 3±0.014 1(3)
ED	0.692 7±0.010 5(9)	0.654 1±0.012 1(9)	0.769 6±0.005 2(9)	0.768 6±0.005 7(10)	0.629 8±0.014 2(8)	0.974 8±0.007 3(10)
BC	0.698 2±0.010 2(2)	0.659 8±0.012 3(2)	0.777 9±0.004 0(2)	0.739 5±0.005 0(1)	0.633 4±0.014 2(3)	0.941 2±0.007 1(2)
PCC	0.696 3±0.008 1(4)	0.657 9±0.010 0(4)	0.777 1±0.003 6(3)	0.740 4±0.002 5(2)	0.630 8±0.011 3(7)	0.940 8±0.004 9(1)
MD	0.690 1±0.007 5(10)	0.651 3±0.009 1(10)	0.770 0±0.005 0(8)	0.761 7±0.010 9(6)	0.625 3±0.009 8(10)	0.965 1±0.015 2(5)
Sorensen	0.694 4±0.008 7(5)	0.656 2±0.009 8(5)	0.770 4±0.005 2(7)	0.766 2±0.004 4(8)	0.632 1±0.011 4(4)	0.972 1±0.005 8(9)
Canberra	0.694 4±0.009 0(6)	0.655 9±0.010 2(6)	0.771 3±0.004 5(6)	0.762 9±0.004 8(7)	0.631 5±0.010 2(6)	0.967 3±0.006 2(7)
Lorentzian	0.693 4±0.010 2(8)	0.654 3±0.011 9(8)	0.773 0±0.004 0(5)	0.761 3±0.004 8(5)	0.628 7±0.014 4(9)	0.965 7±0.006 4(6)
Divergence	0.693 6±0.008 5(7)	0.655 2±0.009 4(7)	0.769 1±0.004 7(10)	0.766 5±0.005 0(9)	0.631 6±0.011 1(5)	0.972 0±0.006 4(8)

(c) Movielens706u 数据集. 在该数据集上，本文提出的算法在 3 个指标上占优.

	F1↑	Accuracy↑	Precision↑	MAE↓	Recall↑	RMSE↓
NTRFC	0.620 7±0.002 1(1)	0.577 9±0.003 3(2)	0.759 9±0.002 8(1)	0.712 4±0.004 9(1)	0.524 7±0.003 1(6)	0.918 9±0.005 9(2)
Cosine	0.616 8±0.007 3(6)	0.573 9±0.009 4(7)	0.751 8±0.002 6(4)	0.729 6±0.033 5(3)	0.523 0±0.009 8(7)	0.941 0±0.052 8(3)
ED	0.604 0±0.001 2(8)	0.559 5±0.001 8(8)	0.744 1±0.003 3(7)	0.787 2±0.002 6(9)	0.508 3±0.001 1(8)	1.03 0±0.005 0(10)
BC	0.619 6±0.001 7(4)	0.576 8±0.002 7(4)	0.753 0±0.003 4(3)	0.713 7±0.002 9(2)	0.526 4±0.003 1(5)	0.916 9±0.003 2(1)
PCC	0.620 0±0.002 5(2)	0.577 2±0.003 8(3)	0.753 3±0.003 8(2)	0.753 0±0.001 9(4)	0.526 7±0.004 3(4)	0.990 6±0.004 2(5)
MD	0.603 0±0.001 7(10)	0.558 4±0.001 9(10)	0.744 9±0.003 4(6)	0.787 4±0.002 6(10)	0.506 4±0.001 1(9)	1.030±0.005 0(9)
Sorensen	0.619 7±0.002 3(3)	0.578 9±0.002 6(1)	0.739 1±0.004 3(10)	0.780 5±0.002 0(6)	0.533 5±0.003 8(1)	1.019±0.004 6(6)
Canberra	0.616 4±0.001 1(7)	0.574 0±0.002 3(6)	0.741 5±0.003 1(8)	0.780 2±0.006 5(5)	0.527 4±0.001 6(3)	1.02 1±0.011 1(7)
Lorentzian	0.603 7±0.001 4(9)	0.559 1±0.001 8(9)	0.747 3±0.003 5(5)	0.782 3±0.002 5(8)	0.506 4±0.001 8(10)	1.024±0.004 7(9)
Divergence	0.617 6±0.002 5(5)	0.575 6±0.003 2(5)	0.740 1±0.005 0(9)	0.782 1±0.002 1(7)	0.530 0±0.004 8(2)	1.022±0.004 6(8)

(d) Eachmovie 数据集. 在该数据集上，本文提出的算法在 3 个指标上占优.

	F1↑	Accuracy↑	Precision↑	MAE↓	Recall↑	RMSE↓
NTRFC	0.654 4±0.000 8(1)	0.618 5±0.000 8(1)	0.759 3±0.000 6(5)	0.785 6±0.001 7(4)	0.575 0±0.001 0(1)	1.007±0.002 5(7)
Cosine	0.653 2±0.000 8(2)	0.616 9±0.000 9(2)	0.761 6±0.000 8(1)	0.775 8±0.000 9(1)	0.571 8±0.001 2(3)	0.979 5±.0001 3(1)
ED	0.631 0±0.000 5(9)	0.592 6±0.000 5(9)	0.754 1±0.000 7(9)	0.808 5±0.000 9(10)	0.542 4±0.000 6(9)	1.016±0.001 4(8)
BC	0.653 1±0.000 8(3)	0.616 9±0.001 0(3)	0.761 2±0.000 7(2)	0.776 4±0.000 9(2)	0.571 9±0.001 1(2)	0.980 4±0.001 3(2)
PCC	0.648 8±0.010 7(4)	0.612 1±0.011 8(4)	0.760 2±0.004 1(3)	0.783 3±0.015 4(3)	0.566 0±0.013 9(4)	0.987 4±0.017 6(3)
MD	0.630 2±0.000 5(10)	0.591 6±0.004 7(10)	0.754 3±0.000 7(8)	0.808 4±0.001 0(9)	0.541 2±0.000 6(10)	1.016±0.001 5(9)
Sorensen	0.634 7±0.000 6(7)	0.596 2±0.000 6(7)	0.756 7±0.000 7(7)	0.800 7±0.000 9(7)	0.546 7±0.000 7(7)	1.007±0.001 4(6)
Canberra	0.635 9±0.000 7(6)	0.597 4±0.000 7(6)	0.756 8±0.000 7(6)	0.799 4±0.000 9(6)	0.548 3±0.000 8(6)	1.006±0.001 4(5)
Lorentzian	0.637 3±0.000 7(5)	0.599 0±0.000 7(5)	0.760 1±0.000 8(4)	0.793 4±0.000 9(5)	0.548 6±0.000 9(5)	0.997 4±0.001 3(4)
Divergence	0.632 4±0.000 5(8)	0.593 6±0.000 4(8)	0.753 1±0.000 6(10)	0.807 7±0.000 9(8)	0.545 0±0.000 7(8)	1.017±0.001 4(10)

注：数据用 $\bar{x}\pm s$ 表示，括号内数字为排名.

4　结　论

本文提出了基于多通道特征向量的新三角距离高效推荐算法. 多通道特征向量能降低算法的时间复杂度，新三角距离能更精准地描述特征向量之间的相似度. 在 4 个真实数据集上的实验结果表明，本文算法

比已有算法在多个指标上表现得更好.

下一步工作拟替换Jaccard系数,使用其他距离公式与三角距离结合,构建新的距离公式.另外,考虑将新三角距离应用到可解释性推荐系统中,以期提升可解释性推荐系统的性能.

参考文献:

[1] EKSTRAND M D. Collaborative Filtering Recommender Systems [J]. Foundations and Trends © in Human-Computer Interaction, 2011, 4(2): 81-173.

[2] ZHANG S C, LI X L, ZONG M, et al. Efficient kNN Classification with Different Numbers of Nearest Neighbors [J]. IEEE Transactions on Neural Networks and Learning Systems, 2018, 29(5): 1774-1785.

[3] ZHANG S, LIU L X, CHEN Z L, et al. Probabilistic Matrix Factorization with Personalized Differential Privacy [J]. Knowledge-Based Systems, 2019, 183: 104864.

[4] WEI J, HE J H, CHEN K, et al. Collaborative Filtering and Deep Learning Based Recommendation System for Cold Start Items [J]. Expert Systems With Applications, 2017, 69: 29-39.

[5] STRANG G. The Discrete Cosine Transform [J]. SIAM Review, 1999, 41(1): 135-147.

[6] HU J Y, GAO Z W, PAN W S. Multiangle Social Network Recommendation Algorithms and Similarity Network Evaluation [J]. Journal of Applied Mathematics, 2013, 2013: 248084.

[7] RICCI F, ROKACH L, SHAPIRA B. Introduction to Recommender Systems Handbook [M]. Springer: Recommender Systems Handbook, 2011: 1-35.

[8] LI R Z, ZHONG W, ZHU L P. Feature Screening via Distance Correlation Learning [J]. Journal of the American Statistical Association, 2012, 107(499): 1129-1139.

[9] ZHANG H R, MIN F, ZHANG Z H, et al. Efficient Collaborative Filtering Recommendations with Multi-Channel Feature Vectors [J]. International Journal of Machine Learning and Cybernetics, 2019, 10(5): 1165-1172.

[10] ZHANG H R, MIN F, SHI B. Regression-Based Three-Way Recommendation [J]. Information Sciences, 2017, 378: 444-461.

[11] QIAN G, SURAL S, GU Y L, et al. Similarity Between Euclidean and Cosine Angle Distance for Nearest Neighbor Queries [C] // Proceedings of the 2004 ACM Symposium on Applied Computing. New York: ACM Press, 2004: 1232-1237.

[12] PATRA B K, LAUNONEN R, OLLIKAINEN V, et al. A New Similarity Measure Using Bhattacharyya Coefficient for Collaborative Filtering in Sparse Data [J]. Knowledge-Based Systems, 2015, 82: 163-177.

[13] CHANG D J, DESOKY A H, MING O Y, et al. Compute Pairwise Manhattan Distance and Pearson Correlation Coefficient of Data Points with GPU [C] //2009 10th ACIS International Conference on Software Engineering, Artificial Intelligences, Networking and Parallel/Distributed Computing. Daegu, Korea (South): IEEE, 2009: 501-506.

[14] JIA X Y, LI W W, LIU J Y, et al. Label Distribution Learning by Exploiting Label Correlations [C] // Proceedings of the 32nd AAAI Conference on Artificial Intelligence. Menlo park, CA, AAAI, 2018: 3310-3317.

[15] GENG X, HOU P. Pre-Release Prediction of Crowd Opinion on Movies by Label Distribution Learning [C] //IJCAI'15: Proceedings of the 24th International Conference on Artificial Intelligence, 2015: 3511-3517.

[16] CHA S H. Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions [J]. International Journal of Mathematical Models and Methods in Applied Sciences, 2007, 1(4): 300-307.

[17] DEZA E, DEZA M M. Distances in Geometry [M] //Dictionary of Distances. Amsterdam: Elsevier, 2006: 62-80.

[18] COX T, COX M. Multidimensional Scaling [M]. Chapman and Hall/CRC, 2000.

[19] WANG J, DE VRIES A P, REINDERS M J T. Unified Relevance Models for Rating Prediction in Collaborative Filtering [J]. ACM Transactions on Information Systems, 2008, 26(3): 1-42.

[20] SARWAR B, KARYPIS G, KONSTAN J, et al. Item-Based Collaborative Filtering Recommendation Algorithms [C] // Proceedings of the Tenth International Conference on World Wide Web. New York: ACM Press, 2001: 285-295.

[21] WILLMOTT C J, MATSUURA K. Advantages of the Mean Absolute Error (MAE)over the Root Mean Square Error (RMSE) in Assessing Average Model Performance [J]. Climate Research, 2005, 30: 79-82.

[22] BERGER A, GUDA S. Threshold Optimization for F Measure of Macro-Averaged Precision and Recall [J]. Pattern Recognition, 2020, 102: 107250.

责任编辑 周仁惠