基于兴趣相似度传递的增强 LSH 统计预测算法

夏小娜[1,2] 邹 麒[2]

[1](曲阜师范大学统计学院 山东 曲阜 273165)

[2](曲阜师范大学信息科学与工程学院 山东 日照 276826)

摘 要 随着在线用户和物品数量的不断增长,有必要通过追踪和筛选历史数据,为用户提供机制可参考的决策建议。构建统计预测算法是实现启发式预测用户兴趣的有效机制。因此,在充分利用用户自身历史偏好和潜在偏好的前提下,提出兴趣相似度传递思想,分析用户的社交关联强度,计算用户的邻近社交兴趣和选择趋向特征,设计并实现了可扩展的局部敏感哈希(Improved Local Sensitivity Hashing,ILSH)统计预测算法。实验表明,该算法在有利于相似度计算量剧增的背景下,在提高兴趣预测的准确性和可靠性方面优于其他近似算法。

关键词 个性化预测 增强局部敏感哈希 兴趣相似度传递 潜在偏好 协同过滤 统计预测算法

中图分类号 TP311 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2020.03.048

AN IMPROVED LSH STATISTICAL PREDICTION ALGORITHM BASED ON INTERESTED SIMILARITY PROPAGATION

Xia Xiaona[1,2] Zou Qi[2]

[1](*School of Statistics*, *Qufu Normal University*, *Qufu 273165*, *Shandong*, *China*)

[2](*School of Information Science and Engineering*, *Qufu Normal University*, *Rizhao 276826*, *Shandong*, *China*)

Abstract As the number of online users and items continues to grow, it is necessary to track and sift through historical data to provide mechanisms for users to make informed decisions. Constructing statistical prediction algorithm is an effective mechanism for heuristic prediction of user interest. Therefore, on the premise of making full use of the user's own historical and potential preferences, this paper proposes the idea of interest similarity transfer, analyzes the intensity of user's social association, calculats the user's neighboring social interest and the choice of trend features, and designs and implements ILSH (improved local sensitivity hashing) statistical prediction algorithm. Experiments show that the proposed algorithm is better than other approximation algorithms in improving the accuracy and reliability of interest prediction in the context of increasing similarity computation.

Keywords Personalized prediction Improved local sensitivity hashing Interested similarity propagation Potential preferences Collaborative filtering Statistical prediction algorithm

0 引 言

个性化推荐是大数据和在线应用的关键技术。通过捕捉用户的行为偏好与目标需求,自主为用户提供合适的服务。这里涉及两个需求背景:一是用户知道自己需要的具体是什么,二是用户并不知道。这两方面都需要有效的推荐策略。推荐时需要综合考虑用户自身的潜在需求,也要考虑受邻近用户的影响。但无论是面对怎样的需求背景,当大量的服务结果呈现在用户面前时,用户并不能客观评估排序潜在的大量可供选择的服务。基于统计预测的推荐策略帮助用户从海量候选结果中提供有用和有效的建议,或者做有意义的引导和启发。

收稿日期:2019-04-19。山东省自然科学基金项目(ZR2016FM45);山东省社会科学规划研究项目(19CZKJ08);山东省研究生教育优质课程项目(SDYKC18079);山东省研究生教育创新计划项目(SDYY16092);曲阜师范大学交叉学科研究项目(XKJJC201901)。**夏小娜**,副教授,主研领域:大数据,学习分析,服务计算。**邹麒**,本科生。

协同过滤是实现个性化统计、预测和推荐常用的方法之一,它对群体进行搜索,从中找出与用户兴趣偏好近似的其他用户作为“兴趣近邻”,对近邻所偏好的相关内容进行分析和考察,将它们组合起来,计算近似度和推荐权重,构造出排序的候选列表[1]。

在对大数据集进行分析并生成推荐列表时,基于物品的过滤推荐方法明显要比基于用户的更快,但存在维护物品相似度表的额外开销。基于物品的推荐和基于用户的推荐,对于数据集的稀疏性处理存在差异,但算法的本质是类似的[2-3],在搜索邻近用户时,以基于用户或物品的相似度为基础,所实现的推荐效果取决于相似度度量的准确性和有效性,以及关于相似度空间搜索和计算过程的能力。邻近用户的“邻近”体现为用户间关于目标的近似选择,以此所体现的近似兴趣偏好,是围绕用户兴趣借助算法实现的用户邻近域界定,即为“兴趣近邻”[4-5]。

有关在线平台的服务推荐,无论是基于用户的推荐还是基于物品的推荐,相关的评分向量都是高维的,在高维数据空间中做到快速地定位相似的用户或者物品,一般情况下有两种解决思路:最近邻域检索和近似最近邻检索。因检索过程是个 NP 问题,因此通常采取近似最近邻方式。LSH(Locality-Sensitive Hash)[6]就是其中有效的研究方法。LSH 的含义表征为将高维空间中邻近的点 HASH 映射到低维空间后仍距离较近,原本较远的点映射后仍具有较远的距离[7]。本文改进 LSH 应用模型,充分计算用户需求域的邻近关联信息,结合用户自身的潜在偏好趋向,构造自主的统计预测机制,设计了 ILSH 算法。

1　相关工作

基于分布式敏感哈希,设计隐私保护和可拓展的服务推荐方法[8]。有效降低实时推荐的运算复杂度,提高评分数据在高维空间中的相似性查询效率;通过运用两个 LSH 策略分类高维数据[13],加入增量聚类算法,批量合并相似度矩阵以合并离线聚类算法;LSH 两级结构可以提高检索效率[15],将提取的特征外包给云服务器还需要做深入的研究,以便于减轻数据所有者和数据用户的负担;进一步,使得 LSH 用于视频的异常检测方法[17],将正常活动散列到多个特征桶中,过滤异常活动,以此实现在线更新程序融入适应视频场景变化的 LSH 框架。

基于邻近搜索机制,可实现多对象优化算法和局部敏感哈希的协同,解决“一对多”“多对一”动态选择和传递问题[9-10]。利用 LSH 发现真正感兴趣的事件,加快集群发现过程,保持聚类质量[11]。但该方法并没有应用于多个社交媒体数据集,无法比较同一事件在多个平台的效果,方法还需要充分检验以扩展到更复杂的情形。

LSH 在 Web 服务中也得到了有效运用。针对 MapReduce中聚类大规模数据集时的有效分布式密度峰值问题[12],设计 LSH 进化算法,以支持用户指定所期望的近似准确度,减少清洗数据和计算消耗;设计基于 MapReduce 编程模型的 LSH 并行集合相似度关联方法,可以减少计算相似度时需求比较的次数;实现 LSH 在 WoS(Web of Science)和 Scopus 匹配中的应用[16],实现检测精确匹配,利于衡量高频数据的重叠情况。

SLH 已得到了广泛应用,许多有关 SLH 的研究,多半是局限于某一领域或者特定数据集参数的调优,或者直接用于部分数据的处理[18],并没有从 SLH 数据结构和算法流程上进行调整和改进,在应用过程中,同样也带来了其他没有解决的问题。

SLH 已得到了广泛应用,许多有关 SLH 的研究,多半是局限于某一领域或者特定数据集参数的调优,或者直接用于部分数据的处理[18],并没有从 SLH 数据结构和算法流程上进行调整和改进,在应用过程中,同样也带来了其他没有解决的问题。本文从 SLH 设计结构出发,扩展优化算法,在提高统计预测运算效率的前提下,提高兴趣相似度的有效传递。

2　局部敏感哈希

2.1　基本定义

定义 1　敏感的函数族

给定一族哈希函数,是一个从欧式空间到哈希编码空间的映射。如果以下两个条件都满足,则称此哈希函数为敏感的函数族[12]。

(1) 若 $p \in B(q,r_1)$,则 $P_{rH}[h(q)=h(p)] \geqslant p_1$;

(2) 若 $p \notin B(q,r_2)$,则 $P_{rH}[h(q)=h(p)] \leqslant p_2$。

定义中 B 表示的是以 q 为中心,r_1 或 r_2 为半径的空间,图 1 是该定义的坐标系分布描述。

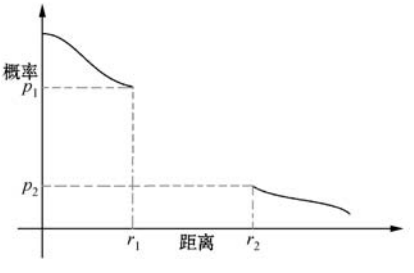

图 1　定义 1 图例

定义 2 给定数据集 S 及相关的局部敏感哈希族 $x,y\in S$,若数据对象 $x,y\in S$ 成立,应满足定义为[13]:

$$P_{h\in H}[h(x)=h(y)]=sim(x,y) \quad (1)$$

上述两种定义的定义角度不同使得这两种定义在形式上差距很大,但是本质上是一致的,即越相近的数据对象发生哈希冲突的概率越高。

2.2 基本思想

在进行预测推荐时,无论是 user-base 还是 item-base,预测过程的相似度计算其本质都是基于物品的评分向量,由用户和评分形成矩阵,具有高维的特点。要实现快速寻求相似的用户或者物品,需要围绕用户或者物品进行近邻检索。

$$\boldsymbol{A}=\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

LSH 的核心理念是通过一组哈希函数,把相似的数据对象哈希到相同的哈希桶中,越相似的对象被哈希到相同桶中的概率越高。这些桶中的数据对象构成目标候选集,从而过滤掉大量的相似概率很低的数据对象。

一般情况下,使用 HASH 技术有效避免冲突,如使用 HashTable 实现与 Redis 的一致性哈希过程。若经过 Hashing 后,两组数据具有相同的 HASH VALUE,则 LSH 认为同两对象是具有相似性的。这样,对每一次近似近邻的查询过程只需要对待检测的对象进行同样的哈希过程,直接从对应的 HASH VALUE 特征桶中找到相似的对象。

建立一个哈希函数族,每个函数随机生成不同的边界,边界之间形成区域。每个函数进行向量运算,生成一条有向线,如图 2 所示,这些有向线的集合即是哈希族。生成有向线的条数是穷举过程,目的是找到一个合适的位置,最终被圈在同一区域的点将视为相邻的。

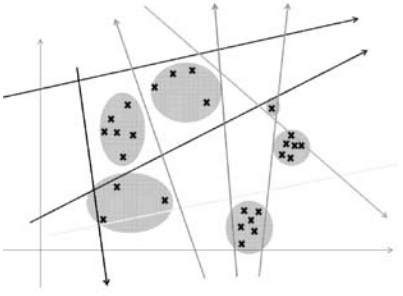

图 2 LSH 算法思想的几何图形解释

LSH 运算快,可以跨平台实现合作推荐,而不破坏用户的隐私。但 LSH 随机分域过程也可能存在错误。解决这个问题有两个思路:一是使用多个独立的哈希表,进行多次区域分割;二是推荐前的多检索策略,对于每一次检索进行评分预测,求取平均。

2.3 增强局部敏感哈希

局部敏感哈希的定义中不难看出,LSH 虽然是近似最优技术,但是并不能保证计算结果的精确性,通过 LSH 我们能得到一个或多个 Hash 表,一次哈希会有很大的可能性将非相似的数据哈希到相同的哈希桶中,这种将非相似数据对象哈希到相同哈希桶中的情形称为纳伪(False Positive)。同时未将真正相似的对象哈希到相同哈希桶中的情形称为拒真(False Negative)。为了保证局部敏感哈希的查询质量,需要尽量降低 False Positive 和 False Negative,也就是实现 LSH 的增强。常用的增强 LSH 的方法有使用多个独立的哈希表,运用 AND、OR、XOR 等操作以及这些操作的级联运算。

增强局部敏感哈希的执行过程体现为:

输入:“user-item”评分记录,pool_size 代表每个哈希表所对应哈希函数的个数,hash_count 是哈希表的个数。

输出:具有 hash_count 个数目的哈希索引结构。

过程:

Step1 算法初始化。将“user-item”评分记录转换为“user-item”评分矩阵,具有 m 个用户、n 个项目的评分矩阵 $\boldsymbol{D}_{m\times n}$ 表征为:

$$\boldsymbol{D}_{m\times n}=\begin{vmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,n} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,n} \\ \vdots & \vdots & & \vdots \\ r_{m,1} & r_{m,2} & \cdots & r_{m,n} \end{vmatrix}$$

对于一个用户,其评分记录可以表示为向量 $\boldsymbol{u}_k=(item_{k,1.q},item_{k,2.q},\cdots,item_{k,m.q})$,其中 $item_{k,l.q}(1\leqslant l\leqslant n,1\leqslant k\leqslant m)$ 表示用户 k 对于物品的评分,若该用户从未评议过该项目则该评分为 0。

Step2 LSH 构造。对于每个用户 $u\in U$,根据既定的哈希函数族 $\{h_k(u)\}$ 将评分记录向量 $\boldsymbol{u}_k$ 映射到哈希空间中。$h_k(u)$ 计算公式表示为:

$$h_k(u)=\begin{cases}1 & \boldsymbol{u}_k\circ\boldsymbol{v}>0\\ 0 & \boldsymbol{u}_k\circ\boldsymbol{v}\leqslant 0\end{cases} \quad (2)$$

式中:$\boldsymbol{v}$ 是一个 m 维向量 $(v_1,v_2,\cdots,v_m)$,$(1\leqslant i\leqslant m)$,其中 v_i 的取值范围为 $[-1,1]$,运算符 $\circ$ 表示对两个向量进行点积运算。

对于式(2)可以用一下物理模型进行描述:

向量 $\boldsymbol{v}$ 是一个对高维空间进行分割的超平面,LSH 的过程即是将分布在高维评分空间中的用户评分点集

进行区域划分,基于前文中关于 LSH 基本思想的阐述,如果两个点相似,则会有极高的概率被超平面划分到同一个区域当中。

Step 3　LSH 索引构建。对于 $\boldsymbol{D}_{m\times n}$ 中的每一个评分向量 $\boldsymbol{u}_k=(item_{k,1.q},item_{k,2.q},\cdots,item_{k,m.q})$,利用哈希函数进行映射,并对每个哈希表进行分桶构建索引。

Step 4　用户的兴趣相似性检索。有关用户的兴趣相似性检索,只需将 hash_count 个哈希表中处于一个桶中的所有用户的并集作为待预测目标用户的兴趣最近邻集合。在此基础上,运用上述三步,计算目标最邻近用户的在线哈希族函数个数 N,找到规模等于 N 的哈希桶,将用户作为相似的候选“近邻”放进哈希桶。

3　ILSH 统计预测算法

现实中的推荐系统的物品远远多于用户,而对于单个用户而言,其评分向量往往是极其稀疏的。单纯通过 LSH 算法找到目标用户的相似用户进而对所有物品进行无差别的加权相加预测评分,并没有考虑到用户会对特定的产品存在一定的爱好偏差这一基本消费心理。因此,本文的 ILSH 只将相似用户对相似物品进行平均加权取值,用以描述用户对特定物品的爱好偏差。

设定待预测目标用户 u 的最近邻集合表示为 U,目标物品 i 的最近邻集合为 I,R_{vi} 是近邻用户 v 对物品 i 的评分,则用户 u 对 i 的预测评分为:

$$p_{ui}=\frac{\sum_{v\in U}\sum_{i\in I}R_{vi}}{|U|+|I|} \tag{3}$$

基于改进 LSH(后文用 ILSH 表示)的统计预测算法主要分为四个步骤,伪码描述如算法 1 – 算法 4。

算法 1　构建 LSH – family

Hash_count: 哈希表的个数
Pool_size: 每个哈希表中哈希函数的个数,即哈希值
Dimensions: 评分向量维度
$H(\cdot)$: LSH 函数族
For k = 1 **to** Hash_count
　For g = 1 **to** pool_size
　　For l = 1 **to** Dimensions
　　$P_{kgl}=random[-1,1]$
　　End For
　　$\boldsymbol{v}_k=(P_{kg1},P_{kg2},\cdots,P_{kgdimensions})$
　EndFor
$H_k=(\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_{pool\_size})$
EndFor

算法 2　分别对 user 和 item 构建索引

$\boldsymbol{u}(j)$:用户 j 的评分向量
$\boldsymbol{i}(k)$:物品 k 的评分向量
$H_u(\cdot)$:根据用户评分向量建立的 LSH-family
$H_i(\cdot)$:根据物品评分向量建立的 LSH-family
Users:用户列表
Items:为物品列表
/ * 根据用户评分向量和物品评分向量建立不同维度的 LSH-family 和哈希表 * /
For $\boldsymbol{u}(j)$ **in** Users
　$H_j(\boldsymbol{u}(j))=(\boldsymbol{v}_1\cdot\boldsymbol{u}(j),\boldsymbol{v}_2\cdot\boldsymbol{u}(j),\cdots)$
　$U_{\text{bucket}}[H_j(\boldsymbol{u}(j))]\cdot append(user_j)$
For $\boldsymbol{i}_{(k)}$ **in** Items
　$H_k(\boldsymbol{i}_{(k)})=(\boldsymbol{v}_1\cdot\boldsymbol{i}(k),\boldsymbol{v}_2\cdot\boldsymbol{i}(k),\cdots)$
　$I_{\text{bucket}}[H_k(\boldsymbol{i}_{(k)})]\cdot append(item_k)$

算法 3　相似性检索

U_{target}: 目标用户
I_{target}: 目标物品
For x = 1 **to** Hash_count
　$hvu=H_u(U_{\text{target}})$
　$uset+=U_{\text{bucke}}[hvu]$
EndFor
For y = 1 **to** Hash_count
　$hvi=H_i(I_{\text{target}})$
　$Iset+=I_{\text{bucket}}[hvi]$
Endfor
EndFor

算法 4　统计预测评分

$similar\_ratings=rating[uset,:]$
$similar\_ratings=similar\_ratings[:,Iset]$
$p\_rating=similar\_ratings$
$[similar\_ratings.nonzero()]mean()$

4　实　验

4.1　训练数据集及评价指标

ILSH 算法的实验数据集选自 University of Minnesota 的 GroupLens 课题组提供 MovieLens(http://grouplens.org/datasets/movielens/),涉及 6 040 个用户关于 3 706 部电影的评议投票,共包括 1 000 000 个评分记录,设定电影的评分数值分布在区间[0, 5]。评分越高,则表明某用户对某电影的兴趣偏好度越大。

有关用户的评分统计度量标准定位于平均绝对误差(Mean Absolute Error, MAE),通过计算兴趣偏好近似用户的评分与待评估的目标用户评分之间的偏差,根据此偏差大小预测准确性。平均绝对误差的数值越

小,推荐准确性越高。假设待预测的用户评分表示为向量表达式 $\boldsymbol{p}=(p_1,p_2,\cdots,p_m)$,实际的用户评分向量为 $\boldsymbol{r}=(r_1,r_2,\cdots,r_m)$,则平均绝对误差为:

$$MAE=\frac{\sum_{i=1}^{m}|p_i-r_i|}{m} \tag{4}$$

4.2 实验设计及结果分析

4.2.1 pool_size 和 hash_count 的选择

为了提高哈希的查询质量,尽量降低 False Positive 和 False Negative,在实施中,通常会采用两种策略[14]:

(1) 在一个 Hash 表内使用更多的哈希函数;

(2) 建立多个 Hash 表。

本文实验主要考察 pool_size 和 hash_count 两个参数对实验结果的影响,训练过程如图 3 - 图 5 所示。用 PYTHON 实现本文算法,从实验数据的训练结果中可以看出,当相关参数值分别设定为 pool_size = 7, hash_count = 12, 所得到的 MAE 结果值最低,也是最好的实验结果。通过取不同的参数值训练数据可以看出,对于兴趣度相似或近似的用户,可以取得更好的检索准确率,从而确保较高的目标服务推荐质量,提高用户的满意度。

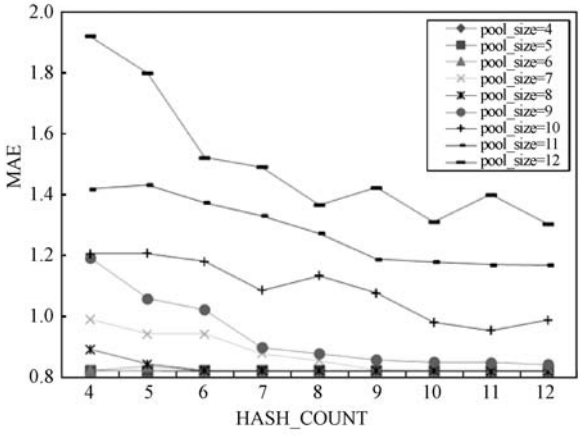

图 3 不同参数对 MAE 的影响

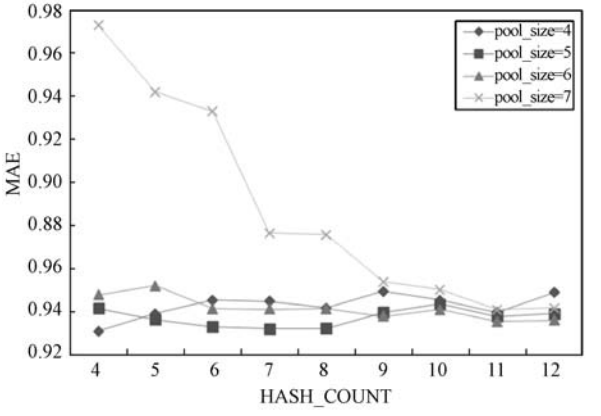

图 4 不同参数对 MAE 的影响

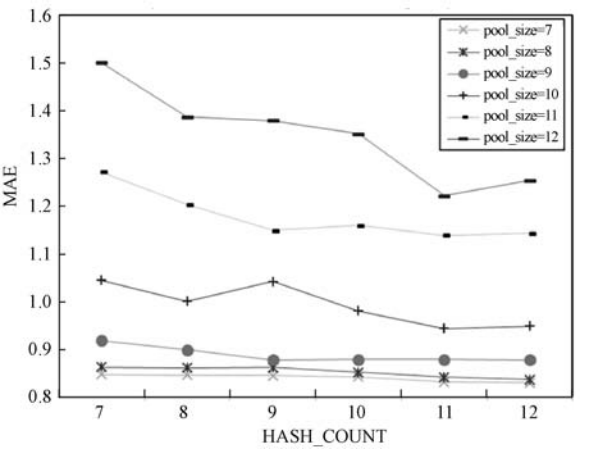

图 5 不同参数对 MAE 的影响

4.2.2 与传统基于用户/物品的协同过滤算法的对比

为检验 ILSH 算法的优势,在算法训练中,我们划定不同用户评分的稀疏数据矩阵展开针对性检验。稀疏数据并非无用数据,数据的稀疏度指的是不存在数据的多维结构的单元的相对百分比,在数据稀疏的前提下从多维结构中能否学习并挖掘出更多有效数据,是算法的一个重要衡量指标。这里我们设定稀疏度计量区间为[0.100,0.300],步长为 0.025,在近似用户评分矩阵的删除数据比例上,比较用户评分矩阵的稀疏度所实现的数据获取效果,即对比本文提出的 ILSH 与传统的基于用户 PCC(User-based Pearson Correlation Coefficient, UPCC)的协同过滤算法和基于物品(Item-based Peason Correlation Coefficient, IPCC)的协同过滤算法的 MAE 值。根据 4.2.1 节的实验结果,设定 hash_count = 12, pool_size = 7。

实验结果如图 6 所示,当数据稀疏度逐渐上升时,UPCC、IPCC、ILSH 的 MAE 都会有一定程度的上升,但本文提出的 ILSH 算法的 MAE 一直处于较小的水平,且比传统的 UPCC 和 IPCC 的 MAE 低很多。

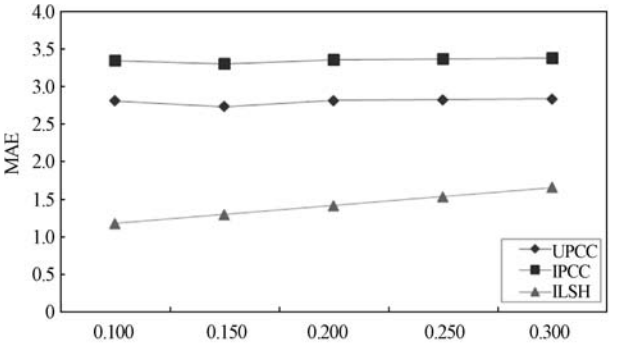

图 6 UPCC、IPCC、ILSH 的 MAE 随着数据稀疏度变化的对比

4.2.3 ILSH 与最新 LSH 改进算法的比较

为将 ILSH 同最新 LSH 算法的执行结果进行有效

比较,采用了与 4.2.1 节相同的实验方案,选取不同的用户评分矩阵稀疏度,比较用户评分矩阵在不同的稀疏度下两种算法的 MAE,实验对比结果如图 7 所示。

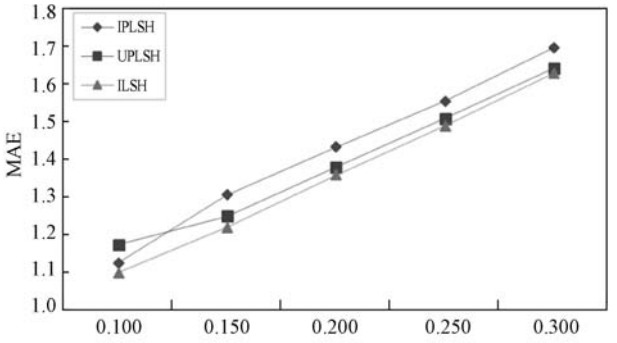

图 7　相关 LSH 算法与 ILSH 算法的 MAE 应对数据稀疏度的变化对比

从图 7 中可以看出,本文提出的 ILSH 算法的 MAE 值一直处于较低的水平且一直比 IPLSH 和 UPLSH 要低。这验证了 ILSH 算法的相对稳定性和准确性。

5　结　语

针对物品推荐体系中高维的用户评分数据以及物品选择和管理的数据稀疏性对推荐决策带来的影响,本文基于最新 LSH 优化算法提出了基于 ILSH 的统计预测算法。实验表明,本文提出的 ILSH 能很好地应对海量高维数据近似计算,有效减少数据稀疏度对统计预测精度的影响。今后将基于兴趣偏好的“近邻的近邻也是我的近邻”这一理论,结合机器学习算法提高用户兴趣统计预测结果的准确性和智能性。

参　考　文　献

[1] Segaran T. Programming collective intelligence [M]. O'Reilly Media, 2007:8 - 9.

[2] Qian J, Huang Z, Qiang Z, et al. Hamming metric multi-granularity locality-sensitive bloom filter[J]. IEEE/ACM Transactions on Networking, 2018, 26(4):1660 - 1673.

[3] Wang Q, He M, Du M, et al. Searchable encryption over feature-rich data[J]. IEEE Transactions on Dependable and Secure Computing, 2016,15(3):496 - 510.

[4] Cohen Y, Hendler D. Scalable detection of server-side polymorphic malware [J]. Knowledge-Based Systems, 2018, 156:113 - 128.

[5] Guo C, Tian P, Chang C C. A Privacy preserving weighted similarity search scheme for encrypted data[J]. IET Information Security, 2018, 13(1):61 - 69.

[6] Gionis A. Similarity search in high dimensions via hashing [C]// Proceedings of the 25th International Conference on Very Large Data Bases. Morgan Kaufmann Publishers Inc., 1999: 518 - 529.

[7] Feng X K, Cui J T, Li H, et al. An efficient LSH indexing on discriminative short codes for high-dimensional nearest neighbors[J]. Multimedia Tools and Applications, 2019,78(17): 24407 - 24429.

[8] Qi L, Zhang X, Dou W, et al. A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data[J]. IEEE Journal on Selected Areas in Communications, 2017, 35(11):2616 - 2624.

[9] Sohrabi M K, Azgomi H. Parallel set similarity join on big data based on Locality-Sensitive Hashing[J]. Science of Computer Programming, 2017,145:1 - 12.

[10] Xia Z, Zhu Y, Sun X, et al. Towards privacy-preserving content-based image retrieval in cloud computing[J]. IEEE Transactions on Cloud Computing, 2015,6(1):276 - 286.

[11] Abdulhayoglu M A, Thijs B. Use of locality sensitive hashing (LSH) algorithm to match Web of Science and Scopus [J]. Scientometrics, 2018,116(2):1229 - 1245.

[12] Cormode G, Dasgupta A, Goyal A, et al. An evaluation of multi-probe locality sensitive hashing for computing similarities over web-scale query logs[J]. PloS ONE, 2018, 13(1):e0191175.

[13] Charikar M S. Similarity estimation techniques from rounding algorithms[C]// Proceedings of the thiry-fourth annual ACM symposium on Theory of computing. ACM, 2002: 380 - 388.

[14] Zhu Z, Xiao J, He S, et al. A multi-objective memetic algorithm based on locality-sensitive hashing for one-to-many-to-one dynamic pickup-and-delivery problem[J]. Information Sciences An International Journal, 2016, 329(C):73 - 89.

[15] Kaleel S B, Abhari A. Cluster-discovery of Twitter messages for event detection and trending[J]. Journal of Computational Science, 2015, 6:47 - 57.

[16] Zhang Y, Cheny S, Yu G. Efficient distributed density peaks for clustering large data sets in MapReduce[C]// 2017 IEEE 33rd International Conference on Data Engineering (ICDE). IEEE, 2017:3218 - 3230.

[17] Zhang Y, Lu H, Zhang L, et al. Video anomaly detection based on locality sensitive hashing filters[J]. Pattern Recognition, 2016, 59(C):302 - 311.

[18] Leskovec J, Rajaraman A, Ullman J D. Mining of massive datasets[M]. Cambridge University Press, 2014.