第 43 卷 第 11 期 计 算 机 工 程 2017 年 11 月
Vol. 43 No. 11 Computer Engineering November 2017

·人工智能及识别技术· 文章编号：1000-3428(2017)11-0193-05 文献标志码：A 中图分类号：TP391

基于用户近邻的 *N* 维张量分解推荐算法

陈健美，孙亚军

（江苏大学 计算机科学与通信工程学院，江苏 镇江 212013）

摘 要： 基于张量分解的推荐算法存在推荐精度较低和数据稀疏的问题。为此，在传统的张量分解模型基础上，引入用户近邻信息，提出一种新的 *N* 维张量分解算法。利用上下文感知信息，把隐式反馈信息作为张量的第 3 维度，以建立 *N* 维张量分解模型，为进一步提高推荐质量，加入用户近邻信息来优化 *N* 维张量分解模型，以提高张量分解推荐算法的准确率。实验结果表明，融合用户近邻的张量分解推荐算法比传统的张量分解算法具有更好的准确性，能有效解决稀疏性和准确性问题。

关键词： 协同过滤算法；反馈信息；主成分分析；张量分解；推荐算法

中文引用格式： 陈健美，孙亚军. 基于用户近邻的 *N* 维张量分解推荐算法[J]. 计算机工程，2017，43(11)：193-197.

英文引用格式： CHEN Jianmei，SUN Yajun. *N* Dimensional Tensor Decomposition Recommendation Algorithm Based on User's Neighbors[J]. Computer Engineering，2017，43(11)：193-197.

N Dimensional Tensor Decomposition Recommendation Algorithm Based on User's Neighbors

CHEN Jianmei，SUN Yajun

（School of Computer Science and Telecommunication Engineering，Jiangsu University，Zhenjiang，Jiangsu 212013，China）

【Abstract】 Recommendation algorithm based on tensor factorization has low accuracy and data sparseness problem. Therefore，on the basic of the traditional tensor decomposition model，this paper introduces the user nearest neighbor information，and proposes *N* dimensional tensor decomposition model. Using context aware information，it uses implicit feedback information as the third dimension to establish *N* dimensional tensor decomposition model. To further improve the the quality of recommendation，it adds the user nearest neighbor information to optimize the *N* dimensional tensor decomposition model to improve the accuracy of the tensor decomposition recommendation algorithm. Experimental results show that the tensor decomposition recommendation algorithm combined with user nearest neighbor has better accuracy than the traditional tensor decomposition algorithm，can effectively solve the sparsity and accuracy problems.

【Key words】 collaborative filtering algorithm；feedback information；Principal Component Analysis（PCA）；tensor decomposition；recommendation algorithm

DOI： 10.3969/j.issn.1000-3428.2017.11.031

0 概述

随着互联网技术的飞速发展，信息量过载大的问题受到广泛关注，用户购买所需物品困难越来越大，推荐系统应运而生[1]。

基于领域的协同过滤算法在推荐系统领域应用广泛[2]，其主要思想是，利用用户的历史行为来预测用户喜好。该算法存在稀疏性和可扩展性的两大问题，主要是因为用户评分矩阵稀疏和巨大的用户量，导致算法的复杂度呈非线性增长。

基于模型的协同过滤算法常被用于大规模数据集，最典型的方法就是矩阵分解（Singular Value Decomposition，SVD）。矩阵分解方法对用户-项目评分矩阵分解之后，将没有用的数据项去除，从而达到降噪和降低稀疏性的目的，非常适合处理二维稀疏矩阵分解[3-4]。

协同过滤算法和矩阵分解非常适合用来处理二维用户-项目评分矩阵，然而，对于高维度的数据，这些传统的方式通常是将多维转为二维关系进行处理，难免会丢失一些多维之间的整体关系信息。如文献[5]提出将用户-项目-标签三元组数据信息进行降维分解，分解成 3 个维度两两相关的二维矩阵，即用户-项目、-用户标签、项目-标签，然后对这 3 个二维矩阵分解进行矩阵分解。但是，这种方式难免

作者简介： 陈健美（1962—），女，教授、博士，主研方向为模式识别、医学图像处理与分析、数据挖掘；孙亚军，硕士研究生。

收稿日期： 2016-08-01 **修回日期：** 2016-11-13 **E-mail：** cjm@ujs.edu.cn

会丢失一部分整体的关联信息。近年来,解决张量数据分解的数学原理越来越成熟,研究者提出了许多的张量数据分解的方法。文献[6]提出从整体考虑推荐模型,将用户-项目-标签当成一个整体,建立三维张量分解模型,从而实现推荐。这种方式不会丢失整体之间的关联。

文献[7]将上下文信息作为额外的维度加入到二维矩阵模型中,建立起 N 维张量模型,提高了推荐精度。张量分解方法能够处理相对较为稀疏的数据集,但是如何利用张量分解从稀疏的数据集中进一步提高推荐的准确性至关重要。为此,本文提出将上下文感知中的隐式反馈信息作为张量的第 3 维度,以建立 N 维张量分解模型。为了进一步提高推荐质量,引入用户近邻信息来优化 N 维张量分解模型。

1 N 维张量分解模型

1.1 矩阵分解算法

在 Netflix 大赛中,矩阵分解首次受到学术领域的特别关注,其后在推荐系统中得到了广大的应用。矩阵分解利用行列式变换原理对矩阵进行分解,将原有的用户-项目评分矩阵拆分成 2 个相对较小的因子矩阵的乘积[8],即用户因子矩阵和项目因子矩阵,从而将原有评分矩阵降维到 2 个低维度的特征矩阵,再利用这 2 个矩阵相乘得到近似的评分矩阵,即 $\boldsymbol{Y}\approx\boldsymbol{U}\boldsymbol{I}^{\mathrm{T}}$,其中,矩阵 $\boldsymbol{Y}$ 为评分矩阵;矩阵 $\boldsymbol{U}$ 和矩阵 $\boldsymbol{I}$ 是模型参数,可以通过优化目标函数求得:

$$\boldsymbol{L}=\min_{\boldsymbol{UI}}(r_{nm}-\hat{r}_{nm})^2+\lambda(\|\boldsymbol{U}\|^2+\|\boldsymbol{I}\|^2)$$

其中,正则项 $\lambda(\|\boldsymbol{U}\|^2+\|\boldsymbol{I}\|^2)$ 是为了防止过拟合。优化后的目标损失函数如下:

$$\boldsymbol{L}(\boldsymbol{U},\boldsymbol{V})=\sum_{i=1}^{N}\sum_{\mathrm{j}=1}^{M}(\boldsymbol{R}_{nm}-\boldsymbol{U}_i^{\mathrm{T}}\boldsymbol{V}_j)^2+\lambda(\|\boldsymbol{U}\|^2+\|\boldsymbol{I}\|^2)$$

其中,λ 是为了避免过度拟合现象而引入的惩罚因子。矩阵分解的直观图如图 1 所示。

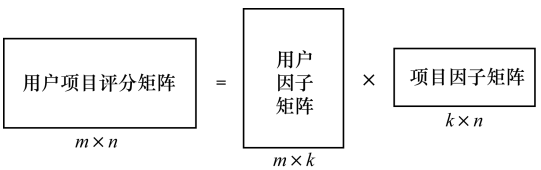

图 1 矩阵分解直观图

1.2 张量分解模型

张量分解是对二维矩阵分解的进一步扩展,因此也称为高阶奇异值分解(Signular Value Decomposition,HOSVD)[9],可以建立比标准的 SVD 分解模型更加精准的模型,因为 HOSVD 算法可以同时实现对张量分解各阶数据进行降维。张量分解的方式主要有 cp 模型、Tucker 模型。Tucker 分解利用主成分分析(Principal Component Analysis,PCA)的方法,将原来的张量近似分为一个更小的核张量与多个因子矩阵的 model-n 乘积。三维张量形式如下:

$$\boldsymbol{Y}=\boldsymbol{G}\times_1\boldsymbol{U}\times_2\boldsymbol{V}\times_3\boldsymbol{C}$$

其中,向量 $\boldsymbol{Y}$ 是三维张量,向量 $\boldsymbol{G}$ 是三维正交张量,矩阵 $\boldsymbol{U}$、$\boldsymbol{V}$、$\boldsymbol{C}$ 是二维因子矩阵,$\times_i$ 是 model-i 张量-矩阵乘法。

张量 $\boldsymbol{Y}$ 的第(i,j,l)个元素为:

$$\hat{y}=\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{l=1}^{k}g_{ijl}u_i\circ v_j\circ c_l$$

张量模型 HOSVD 分解模型如图 2 所示。

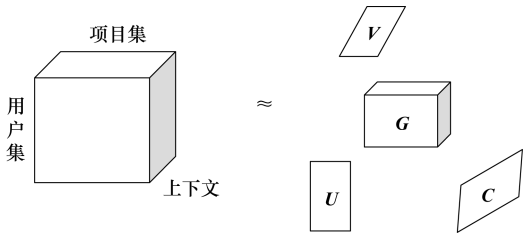

图 2 三维 HOSVD 张量分解模型

2 协同过滤算法与张量分解模型

2.1 基于用户近邻的协同过滤算法

基于用户近邻协同过滤算法最重要的一步是如何根据用户项目评分矩阵 $\boldsymbol{R}$ 计算出用户的相似度 $sim(i,v)$,计算用户之间的相似度的常用方式有如下 3 种:余弦相似性,修正余弦相似性,earson 相似性。

余弦相似性[10-11](Cosine Correlation,CC):用户对 n 个项目的评分可以看成该用户的 n 维向量,2 个 n 维向量的夹角余弦相当于 2 个用户的相似度。

$$sim(i,v)=\frac{\sum_{j\in I_{iv}}\boldsymbol{R}_{ij}\boldsymbol{R}_{vj}}{\|\sum_{j\in I_{iv}}\boldsymbol{R}_{ij}^2\|\ \|\sum_{j\in I_{iv}}\boldsymbol{R}_{vj}^2\|}$$

其中,矩阵$\boldsymbol{R}_{ij}$、$\boldsymbol{R}_{vj}$分别表示 i、v 用户对物品 j 的评分,项目集矩阵$\boldsymbol{I}_{iv}=\{j\in\boldsymbol{I}\mid\boldsymbol{R}_{ij}\neq0\cap\boldsymbol{R}_{vi}\neq0\}$表示用户 i、v 在项目集矩阵 $\boldsymbol{I}$ 上的共同评分。

修正余弦相似度(Adjusted Cosine Correlation,ACC):该算法中加入了每个用户的评分因素,公式定义如下:

$$sim(i,v)=\frac{\sum_{j\in\boldsymbol{I}_{iv}}(\boldsymbol{R}_{ij}-\overline{\boldsymbol{R}_i})(\boldsymbol{R}_{vj}-\overline{\boldsymbol{R}_v})}{\sqrt{\sum_{j\in\boldsymbol{I}_i}(\boldsymbol{R}_{ij}-\overline{\boldsymbol{R}_i})^2}\sqrt{\sum_{j\in\boldsymbol{I}_v}(\boldsymbol{R}_{vj}-\overline{\boldsymbol{R}_v})^2}}$$

其中,矩阵$\overline{\boldsymbol{R}_i}$、$\overline{\boldsymbol{R}_v}$分别表示用户 i、v 的平均评分。

Pearson 相关性[12](Pearson Correlation,PC):通过计算 2 个用户的线性相关程度来计算它们的相似

度,Pearson 相关系数公式如下:

$$sim(u,v)=\frac{\sum_{i\in I_{uv}}(R_{ui}-\overline{R}_u)(R_{vi}-\overline{R}_v)}{\sqrt{\sum_{i\in I_{uv}}(R_{ui}-\overline{R}_u)^2}\sqrt{\sum_{i\in I_{uv}}(R_{vi}-\overline{R}_v)^2}}$$

基于用户协同过滤算法在得出用户之间的相似度之后,从而找出当前用户的最近邻用户 w,然后利用用户 w 的评分来预测该用户对物品的评分值。该模型的公式如下:

$$P_{ij}=\overline{R}_i+\frac{\sum_{v\in N(i)}sim(i,v)\cdot(R_{vj}-\overline{R}_v)}{\sum_{v\in N(i)}|sim(i,v)|}$$

其中,矩阵 P_{ij} 表示用户 i 对未评分项目 j 的预测评分。

2.2　基于上下文感知的 N 维张量分解模型

传统的协同过滤算法[13-14]在计算用户或者项目之间的相似度的时候通常利用显性的用户评分信息,然而当出现新用户或者新物品时,用户的评分信息将非常稀疏,而如果充分利用用户在购买商品时的点击、加入购物车等信息就可以在一定程度上解决稀疏性问题,同时可以给出更高精度的推荐算法。

本文将矩阵分解与上下文感知这一用户隐性反馈信息结合,提出基于上下文的张量分解模型。通过张量分解来补充用户、产品、上下文构成的张量中的缺失评分值。

本文使用 r_{uic} 来表示用户 u 在上下文条件 c 下对产品 i 的评分,其中上下文变量 c 代表多个 context 因素,用 $r_{uic_1},r_{uic_2},\cdots,r_{uic_k}$ 表示用户 u 在 $c_1,c_2,\cdots,c_k$ 上下文下对产品 i 的评分。HOSVD 张量分解将 N 阶张量 $\boldsymbol{Y}$ 分解为一个核心张量和 N 个因子矩阵 mode-n 乘积形式,如下:

$$\boldsymbol{Y}=\boldsymbol{G}\times_1\boldsymbol{U}\times_2\boldsymbol{V}\times_3\boldsymbol{C}$$

每一项元素如下:

$$\hat{y}=\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{l=1}^{k}g_{ijl}u_i\circ v_j\circ c_l$$

扩展到 n 个上下文,N 维张量分解把 N 维张量 $\boldsymbol{Y}$ 近似分解为一个小型核张量 $\boldsymbol{G}$ 与 $(n+2)$ 个因子矩阵的 mode-n 乘积形式,如下:

$$\hat{\boldsymbol{Y}}=\boldsymbol{G}\times_U\boldsymbol{U}\times_V\boldsymbol{V}\times_1\boldsymbol{C}_{K_1}\times_2\boldsymbol{C}_{K_2}\times_3\cdots\times_n\boldsymbol{C}_{K_n}$$

其中,每个元素如下:

$$\hat{y}=\sum_{i}^{n}\sum_{j}^{m}\sum_{f_1}^{K_1}\cdots\sum_{f_n}^{K_n}g_{ijf_1\cdots f_n}u_i\circ v_j\circ c_{f_1}\circ\cdots\circ c_{f_n}$$

通过最小化估计值与观测值之间的误差来求解模型参数,损失函数如下 $\boldsymbol{L}(\hat{\boldsymbol{Y}},\boldsymbol{Y})$:

$$\boldsymbol{L}(\hat{\boldsymbol{Y}},\boldsymbol{Y})=\frac{1}{\|\boldsymbol{G}\|_1}\sum_{i,j,k}\boldsymbol{B}_{ijk}l(\hat{y}_{ijk},y_{ijk})+\frac{\lambda}{2}(\|\boldsymbol{U}\|^2+\|\boldsymbol{V}\|^2+\|\boldsymbol{C}\|^2)$$

其中,后一项为防止过度拟合的正则项。

可以采用随机梯度下降法(Stochastic Gradient Descent,SGD)来优化张量 $\boldsymbol{G}$ 和矩阵 $\boldsymbol{U}$、$\boldsymbol{V}$、$\boldsymbol{C}$。在求目标函数最值的时候可以在每一次迭代中可以通过固定 2 个特征向量改变另一个特征向量进行计算,损失函数中 $\boldsymbol{G}$、$\boldsymbol{U}$、$\boldsymbol{C}$、$\boldsymbol{V}$ 等参数可以利用下面公式计算:

$$\partial_G l(\hat{y}_{ijk},y_{ijk})=\partial_{y_{ijk}}l(\hat{y}_{ijk},y_{ijk})\boldsymbol{U}_{i*}\otimes\boldsymbol{V}_{j*}\otimes\boldsymbol{C}_{k*}$$

$$\partial_{U_{i*}}l(\hat{y}_{ijk},y_{ijk})=\partial_{y_{ijk}}l(\hat{y}_{ijk},y_{ijk})\boldsymbol{G}\times_V\boldsymbol{V}_{j*}\times_C\boldsymbol{C}_{k*}$$

$$\partial_{V_{i*}}l(\hat{y}_{ijk},y_{ijk})=\partial_{y_{ijk}}l(\hat{y}_{ijk},y_{ijk})\boldsymbol{G}\times_U\boldsymbol{U}_{i*}\times_C\boldsymbol{C}_{k*}$$

$$\partial_{C_{i*}}l(\hat{y}_{ijk},y_{ijk})=\partial_{y_{ijk}}l(\hat{y}_{ijk},y_{ijk})\boldsymbol{G}\times_U\boldsymbol{U}_{i*}\times_V\boldsymbol{C}_{j*}$$

计算出 $\boldsymbol{U}$、$\boldsymbol{C}$、$\boldsymbol{V}$、$\boldsymbol{G}$ 的梯度以后,再用下面的公式进行迭代:

$$\boldsymbol{G}\leftarrow\boldsymbol{G}-\eta\lambda\boldsymbol{G}-\eta\partial_G l(\hat{y}_{ijk},y_{ijk})$$

$$\boldsymbol{U}_{i*}\leftarrow\boldsymbol{U}_{i*}-\eta\lambda\,\boldsymbol{U}_{i*}-\eta\partial_{U_{i*}}l(\hat{y}_{ijk},y_{ijk})$$

$$\boldsymbol{V}_{j*}\leftarrow\boldsymbol{V}_{j*}-\eta\lambda\,\boldsymbol{V}_{j*}-\eta\partial_{V_{j*}}l(\hat{y}_{ijk},y_{ijk})$$

$$\boldsymbol{C}_{k*}\leftarrow\boldsymbol{C}_{i*}-\eta\lambda\,\boldsymbol{C}_{i*}-\eta\partial_{C_{i*}}l(\hat{y}_{ijk},y_{ijk})$$

其中,$t\leftarrow t+1$,步长 $\eta\leftarrow\eta+1$,不停地迭代得到最优解,然后带入下式:

$$\hat{\boldsymbol{Y}}=\boldsymbol{G}\times_1\boldsymbol{U}\times_2\boldsymbol{V}\times_3\boldsymbol{C}$$

得到近似评分张量。

2.3　融合用户近邻信息的张量分解模型

本文提出结合用户近邻的张量分解模型,将用户近邻信息加入到张量分解表达式中。该模型如下:

$$\boldsymbol{L}(\hat{\boldsymbol{Y}},\boldsymbol{Y})=\frac{1}{\|\boldsymbol{G}\|_1}\sum_{i,j,k}\boldsymbol{B}_{ijk}l(\hat{y}_{ijk},y_{ijk})+\frac{\lambda}{2}(\|\boldsymbol{U}\|^2+\|\boldsymbol{V}\|^2+\|\boldsymbol{C}\|^2)+\tilde{e}$$

其中,$\tilde{e}$ 为用户近邻信息项;$\tilde{\alpha}=\frac{\delta}{2}\sum_{i=1}^{m}\sum_{v\in N(i)}sim(i,v)\|\boldsymbol{U}_i-\boldsymbol{U}_v\|_F^2$;$\delta$ 越大表示用户近邻越大,反之则越小。

最终的损失函数为:

$$\boldsymbol{L}(\hat{\boldsymbol{Y}},\boldsymbol{Y})=\frac{1}{\|\boldsymbol{G}\|_1}\sum_{i,j,k}\boldsymbol{B}_{ijk}l(\hat{y}_{ijk},y_{ijk})+\frac{\lambda}{2}(\|\boldsymbol{U}\|^2+\|\boldsymbol{V}\|^2+\|\boldsymbol{C}\|^2)+\frac{\delta}{2}\sum_{i=1}^{m}\sum_{v\in N(i)}sim(i,v)\|\boldsymbol{U}_i-\boldsymbol{U}_v\|_F^2$$

为求得最优解,需要利用上文中提到的梯度下降法来最小化损失函数 $\boldsymbol{L}(\hat{\boldsymbol{Y}},\boldsymbol{Y})$。在利用梯度下降法进行计算时,相似度的阈值为 $\tilde{\theta}$,当 $sim(i,v)\geqslant\tilde{\theta}$,表示不考虑和用户 i 相似度小于 $\tilde{\theta}$ 的近邻用户。

具体算法步骤为:

输入 用户在上下文下对项目的评分矩阵 $\boldsymbol{A}_{uti}$,用户关系矩阵 $\boldsymbol{F}$

输出 推荐列表

步骤 1 选取上下文信息中合适的维度作为 HOSVD 的新维度。

步骤 2 将用户关系矩阵 $\boldsymbol{F}$ 以及 $\boldsymbol{A}_{uti}$ 代入 2.1 节中的算法计算出用户相似度关系矩阵。

步骤 3 将用户相似度关系代入 2.3 节的损失函数中,利用梯度下降法最小化损失函数。

步骤 4 在某些维度下,为用户推荐在该维度下的前 N 个较大的项目。

2.4 模型复杂度

1.2 节中融合上下文感知 N 维张量分解模型的时间复杂度为 $O(nm\prod_{l=1}^{d}k_l)$,随着上下文维数的增加该模型复杂度呈指数增长。

为了防止 $\boldsymbol{L}(\hat{\boldsymbol{Y}},\boldsymbol{Y})$ 过度拟合,使用范数进行正则化:

$$\boldsymbol{L}(\hat{\boldsymbol{Y}},\boldsymbol{Y})=\frac{1}{\|\boldsymbol{G}\|_1}\sum_{i,j,k}\boldsymbol{B}_{ijk}l(\hat{y}_{ijk},y_{ijk})+\frac{\lambda}{2}(\|\boldsymbol{U}\|^2+\|\boldsymbol{V}\|^2+\|\boldsymbol{C}\|^2)$$

其中,后一项为防止过度拟合的正则项,用来防止过度拟合,$\|\cdot\|_1$ 是 l_1 的范数,其作用是控制核张量的范围。

最小化该目标函数的方法有很多,此处,可以使用随机梯度下降法来优化张量 $\boldsymbol{G}$ 和矩阵 $\boldsymbol{U}$、$\boldsymbol{V}$、$\boldsymbol{C}$。为了加快随机梯度下降,得到最优解,可以采用随机梯度下降法求解模型的解,虽然得到的不是精准的全局最优解,但是在大体方向上是全局最优的。同时,引入用户近邻因子,提高算法的准确率。

3 实验结果与分析

3.1 数据集

本文使用 MovieLens 数据集进行实验测试:该数据集包含 943 个独立用户对 1 682 部电影作出的 10 000 次评分数据[15]。本文实验中随机选取 80% 的数据作为训练集,剩余的 20% 作为测试集。

3.2 评价准则

本文使用均方根误差(RMSE)和平均绝对误差(MAE)这 2 个指标来测量所提出的推荐算法的准确性。

RMSE 指标定义如下:

$$R_{\mathrm{RMSE}}=\frac{\sqrt{\sum_{u,i\in R_{\mathrm{test}}}(r_{ui}-\hat{r}_{ui})^2}}{|R_{\mathrm{test}}|}$$

其中,r_{ui}、$\hat{r}_{ui}$ 分别表示用户 u 对物品 i 的实际评分与预测评分,R_{test} 表示当前用户评分数目。如果 RMSE 越小,那么该算法预测评分的准确度就越高。

MAE 定义如下:

$$M_{\mathrm{MAE}}=\frac{\sum_{u,i\in T}|r_{ui}-\hat{r}_{ui}|}{|R_{\mathrm{test}}|}$$

其中,算法的参数同上。MAE 值越小,算法预测评分的准确度越高。

3.3 结果分析

在本文的算法模型中,有 3 个重要的参数,分别为上下文维度 l、用户近邻影响因子 δ 和相似度阈值,同时将传统的张量分解算法与本文提出的算法进行比较。

1)上下文维度 l 对算法的影响

在比较本文所提算法与传统的张量分解算法的性能时,使用相同的训练集和测试集,计算这 2 个算法的平均绝对误差 MAE 来验证它们的优劣。

设置相同的参数:学习速率 $\alpha=0.001$,约束因子 $\delta=0.01$,迭代次数 $f=100$,分别结合 3 种相似度计算方法进行实验。

在 UB_C_Tensor 算法中,统一取相似度阈值 $\tilde{\theta}=0.5$,分别在不同维度 $l(l=9,10,\cdots,15)$ 上进行实验。MAE 随维度 l 的变化曲线如图 3 所示。

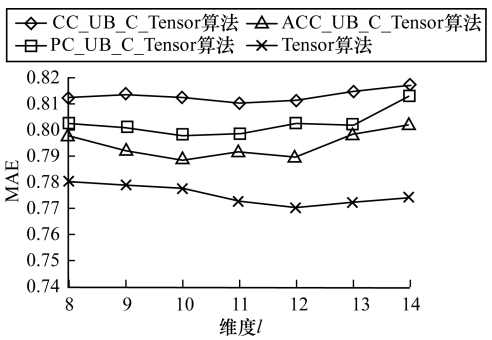

图 3 $f=100$ 时 MAE 随维度 l 的变化曲线

由图 3 可以看出,UB_C_Tensor 的推荐效果比传统张量分解算法要好。同时,使用 cosine 余弦相似度的 CC_UB_C_Tensor 的推荐效果优于其他相似度。

2)用户近邻因子 δ 对算法的影响

用户近邻因子 δ 也会影响算法 UB_C_Tensor。参数设置如下:$\alpha=0.001$,$l=12$,$\lambda=0.01$,$\tilde{\theta}=0.5$,δ

分别为10^{-6}、10^{-5}、10^{-4}、10^{-3}、10^{-2}、10^{-1}、10^{0},使用不同的相似度,对应的 MAE 值随 δ 的变化曲线如图 4所示。

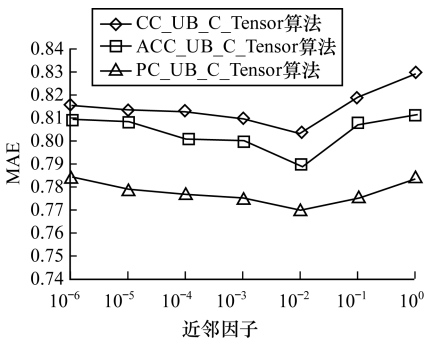

图 4　MAE 随用户近邻因子 δ 的变化曲线

从图 4 可以看出,MAE 的值随着近邻因子 δ 的增大先减小后又变大。这样可以选择合适的 δ 值可以使得用户近邻对张量分解产生更好的推荐效果,在本文的数据集中,δ 取10^{-2}时,推荐效果最优。

3)不同的相似度阈值 $\tilde{\theta}$ 对算法的影响

为了研究不同的相似度阈值对 UB_C_Tensor 算法的影响,将上下文维度设置为 10,除 $\tilde{\theta}$ 之外的其他变量保持跟 2)中实验一致,设置 $\tilde{\theta}$ 分别为 0.1,0.2,…,0.9,结合 3 种相似度计算方式分别代入模型公式计算,由此得到的 MAE 随阈值 $\tilde{\theta}$ 变化的曲线如图 5 所示。

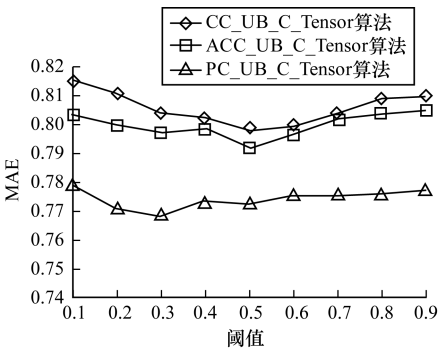

图 5　维度 $l=12$ 时 MAE 随 $\tilde{\theta}$ 的变化曲线

从图 5 可以看出,相似度阈值 $\tilde{\theta}$ 对算法 UB_C_Tensor 有影响,在 3 种相似度计算方法下,随着 $\tilde{\theta}$ 从 0.1 ~0.9 变化,MAE 先减小后增大。选择合适的 $\tilde{\theta}$ 可以获取较好的推荐效果。

4　结束语

本文利用上下文信息构建张量分解模型,讨论上下文维度与推荐结果的关系,并结合用户近邻信息,分析用户近邻因子对相似度阈值的影响。实验结果表明,本文所提出的融合了用户近邻和上下文信息的张量分解推荐算法比传统的张量分解算法具有更好的准确性。下一步将针对大数据环境,利用社交网络信息和其他上下文因素优化张量分解模型,提高推荐实时性。

参考文献

[1] ZHENG Nan,LI Qiudan. A Recommender System Based on Tag and Time Information for Social Tagging Systems[J]. Expert Systems with Applications,2011,38(4):4575-4587.

[2] 陶　俊,张　宁. 基于用户兴趣分类的协同过滤推荐算法[J]. 计算机系统应用,2011,20(5):55-59.

[3] 于洪涛,周　静,张付志. 融合信任传播和矩阵分解的协同推荐算法[J]. 燕山大学学报,2013(5):424-429.

[4] SUN Lifang, LI Shuopeng. Social Tagging Recommendation System Based on K-means Cluster and Tensor Decomposition[J]. Journal of Jiangsu University of Science & Technology,2012,7(7):1-13.

[5] 秦玉珠. 基于社会化标签的个性化推荐算法研究[D]. 哈尔滨:哈尔滨工程大学,2014.

[6] 李　贵,王　爽,李征宇,等. 基于张量分解的个性化标签推荐算法[J]. 计算机科学,2015,42(2):267-273.

[7] 张　爽. 基于张量分解的上下文感知推荐及其应用[D]. 长春:吉林大学,2015.

[8] 李　雪. 基于协同过滤的推荐系统研究[D]. 长春:吉林大学,2010.

[9] 韩　峰,魏国华,吴嗣亮. 基于张量分解的子空间频率估计及应用[J]. 现代电子技术,2010,33(18):138-140.

[10] WANG Licai, MENG Xiangwu, ZHANG Yujie. Context-aware Recommender Systems[J]. Journal of Software,2012,32(3):217-253.

[11] 黄传飞. 基于项目的协同过滤算法的改进[D]. 南昌:江西师范大学,2015.

[12] 曾小波. 基于协同过滤的推荐系统的研究[D]. 成都:电子科技大学,2010.

[13] 祝奇伟,陈家琪. 一种改进相似度计算方法的协同过滤推荐算法[J]. 信息技术,2015(3):13-16.

[14] 项　亮,陈　义,王　益. 推荐系统实践[M]. 北京:人民邮电出版社,2009.

[15] 吴月萍,郑建国. 协同过滤推荐算法[J]. 计算机工程与设计,2011,32(9):3019-3021.

编辑　刘　冰