情报工程 ISSN 2095-915X
第2卷 第4期 30-39 2016年8月

TECHNOLOGY INTELLIGENCE ENGINEERING
ISSN 2095-915X Vol.2 No.4 30-39 Aug 2016

doi:10.3772/j.issn.2095-915x.2016.04.005

推荐系统中的隐语义模型研究

李琳娜，江雪琴

（中国科学技术信息研究所 北京 100038）

摘要：隐语义模型是一种有效的隐含语义分析技术，其核心思想是通过潜在特征联系用户和物品。本文从理论方法的角度，详细介绍了隐语义模型的工作原理、模型表示、学习方法和评价指标等，并通过具体实验分别讨论几种典型隐语义模型算法的推荐效果，以全面认识和理解该模型在推荐系统中的应用特点。

关键词：隐语义模型，推荐系统，隐含语义分析，评价指标

中图分类号：G35，TP39

Research on the Latent Factor Model in Recommender System

LI LinNa，JIANG XueQin

(Institute of Scientific and Technical Information of China, Beijing 100038, China)

Abstract: Latent factor model is an efficient method for latent semantic analysis. The core idea of latent factor model is to link users and items through implicit features. In this paper, the theory of latent factor model was introduced firstly, and its representation and solution was discussed subsequently. Three classic Latent factor models were introduced in details. Furthermore, the performances of these models were compared through experiments based on test dataset, and the characteristic of latent factor model in applications were also discussed.

Keywords: Latent factor model, recommender system, latent semantic analysis, evaluation metrics

基金项目：本文受国家科技支撑计划项目：面向科技情报分析的信息服务资源开发与支撑技术研究（2015BAH25F01），中国工程科技知识中心建设项目：知识组织体系建设（CKCEST-2016-2-10）资助。

作者简介：李琳娜（1981-），博士，研究方向：数据挖掘、个性化推荐。在Ei期刊、中文核心期刊等学术刊物上发表学术论文10余篇，E-amil：liln@istic.ac.cn。江雪琴（1990-），研究方向：推荐系统，知识组织。

1 引言

隐语义模型（Latent Factor Model，简称LFM）是一种有效的隐含语义分析技术，其核心思想是通过潜在特征联系用户和物品。其过程分为三个部分：将物品映射到潜在分类、确定用户对潜在分类的兴趣度、从用户兴趣度高的分类中选择物品推荐给用户[1]。

一般地，推荐系统中用户对物品的行为信息往往表示成评分矩阵的形式，隐语义模型则将这一高维评分矩阵降维到低维度的子空间，使用户和物品的潜在语义关系在子空间中自然显现，从而能很好地描述用户的兴趣偏好，在推荐结果预测方面体现出很高的准确性和稳定性[2]。相类似的模型有pLSA[3]、LSI[4]、LDA[5]等。

隐语义模型有别于传统的协同过滤推荐技术，它不再拘泥于单方面分析用户之间、物品之间的关系，而是通过挖掘用户和物品之间的潜在语义关系以实现推荐，无论是在单模型还是组合模型的推荐中，都具有较好的预测效果。

目前国内单独对隐语义模型的基本思想、工作原理和发展情况做专门分析和系统介绍的成果并不是很多。本文从理论方法的角度，详细介绍了隐语义模型的工作原理、模型表示、学习方法和评价指标等，并通过具体实验分别讨论几种典型隐语义模型算法的推荐效果，以全面认识和理解该模型在推荐系统中的应用特点。

2 隐语义模型

2.1 隐语义模型的形式化定义

隐语义模型（Latent Factor Model，LFM）来源于2006年Netflix Prize大赛中受到广泛关注的奇异值分解（SVD，Singular Value Decomposition），它与PLSA、LDA、隐含类别模型等同属于隐含语义分析技术。隐语义模型的实现基础是矩阵分解，它将用户及物品表示成向量形式，向量的取值由评分模式来确定（一般以显式评分为主），通过矩阵分解将用户和物品映射到维度为F的潜在因子空间，用户－物品的关系则可以表示成该空间上的内积，形成语义上的关联。

假设，给定一个用户－物品的评分数据集R，其中，用户个数为U，物品个数为M，R是一个$U\times M$的矩阵。经隐语义模型建模后，可以得到如图1所示的模型：

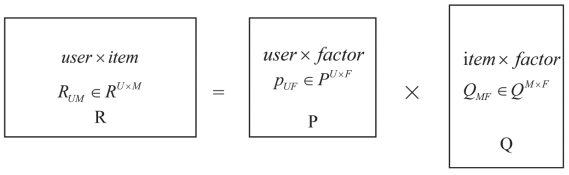

图1 隐语义模型矩阵分解示意图

矩阵P表示用户user对因子类别factor的兴趣度，矩阵Q表示物品item在因子类别factor上的权值，代表物品在因子空间的特征分布。矩阵P和Q的内积表示用户对物品的总体兴趣度值，假设用$\hat{R}_{UM}$表示，具体模型定义如下：

$$\hat{R}_{UM} = P_U Q_M = \sum_{f=1}^{F} p_{Uf} Q_{fM}^{T} \quad \text{(Base-SVD)} \qquad (1)$$

隐语义模型的工作原理可以理解为分类问题，它将用户user对物品item的兴趣偏好划分到个类别，但又不明确到某一类，而是计算其属于每一类的概率，是典型的软分类问题。而且，隐语义模型并不关心分类的角度和粒度，最终的分类结果根据用户的行为数据自动形成，通过设置最终分类数来控制粒度，分类数越大，粒度越细，反之，粒度越粗。隐语义模型从语义分析的角度尽可能的挖掘用户和物品的潜在关联，建立属于用户自己的兴趣偏好特征模型，为最终推荐列表的形成提供了重要基础。

2.2 基于随机梯度下降法的隐语义模型

隐语义模型是基于评分预测进行推荐的算法，其核心问题是矩阵填充，经填充后的评分矩阵可以得到用户对物品的预测评分，推荐模型就可以根据预测评分的排序结果实现推荐。传统的奇异值分解对矩阵填充的代价非常大，所以。通常采用随机梯度下降法，基于最小化 RMSE 构建损失函数，直接通过训练集中的观察值学习 P、Q 矩阵，损失函数定义为：

$$C(p,q)=\frac{1}{2}\sum_{(u,m)\in T}(r_{um}-\sum_{f}p_{uf}q_{mf}^{T})^2 \quad (2)$$

对上述损失函数直接优化容易导致学习的过拟合，因此在损失函数中加入正则项以避免该问题的出现，其中 λ 是正则化参数：

$$C(p,q)=\frac{1}{2}\sum_{(u,m)\in T}[(r_{um}-\sum_{f}p_{uf}q_{mf}^{T})^2+\lambda(\| p_u \|^2+\| q_m \|^2)] \quad (3)$$

采用随机梯度下降法对公式（3）进行优化，通过求参数的偏导找到模型的负梯度，然后迭代不断优化参数。这里，主要有两组参数（p_{uf} 和 q_{mf}），首先对它们求偏导数，可得：

$$\frac{\partial}{\partial p_{uf}}=-(r_{um}-\sum_{f}p_{uf}q_{mf}^{T})q_{mk}+\lambda p_{uk}=-e_{um}*q_{mk}+\lambda p_{uk}$$

同理：

$$\frac{\partial}{\partial q_{mf}}=-(r_{um}-\sum_{f}p_{uf}q_{mf}^{T})p_{uk}+\lambda q_{mk}=-e_{um}*p_{uk}+\lambda q_{mk} \quad (4)$$

$$p_{uf}=p_{uf}+\alpha(e_{um}*q_{mk}-\lambda p_{uk})$$

$$q_{mf}=q_{mf}+\alpha(e_{um}*p_{uk}-\lambda q_{mk})$$

其中，e_{um} 表示观察值与预测值的误差，α 是学习速率，它的取值可通过反复实验获得。而正则项系数 λ 是影响预测结果准确度的重要因素，对于超参数 λ 的选择需要在训练集上反复验证。模型中参数的初始值通常采用随机数生成器进行采样，如采用取值范围为（0，1）的浮点数对参数做初始化等。

3 隐语义模型的改进算法

3.1 Bias–SVD

推荐系统中观察到的用户－物品评分，不仅和用户、物品间的交互相关，而且很大程度上也与用户或物品本身相关，这被称为偏差或偏置（Bias）。例如，一个评分系统有些固有属性与用户或物品无关，而用户也有些属性与物品无关，物品有些属性与用户无关，部分用户总体上给的评分都较高或较低，而部分物品得到的评分总体上也都较高或较低等，这些都可以认为是偏差或偏置。因此，只考虑用户和物品的交互关系并不太合理。相反，尝试通过偏差解释用户或物品本身的属性特征，是对原有隐语义模型的很好补充。偏置项 [6,7] 定义如下：

$$b_{um}=\mu+b_u+b_m \quad (5)$$

b_{um}：评分的偏置项，表示评分系统、用户和物品本身的属性特征；

μ：全局平均数，所有评分对的平均数，代表评分系统本身对用户评分的影响；

b_u：用户偏置（user bias），表示因用户的评分习惯与物品无关部分的因素，如用户喜欢打高分等；

b_m：物品偏置（item bias），表示物品接受的评分中与用户无关部分的因素，如物品本身的质量等。

结合偏置项的定义，预测评分 $\hat{r}_{um}$ 可重新定义为：

$$\hat{r}_{um}=\mu+b_u+b_m+p_uq_m^{T} \quad (6)$$

这里，模型主要包括四个部分，全局平均数、用户偏置、物品偏置以及用户物品间的交互关系，除全局平均数可直接得到之外，其他参数需要通过学习算法求解得到，λ_1、λ_2 为正则项系数。Bias–SVD 学习的目标函数可定义为：

$$C=\min\sum_{(u,m)\in T}[(r_{um}-\mu-b_u-b_m-p_uq_m^{T})^2+\lambda_1(b_u^{\ 2}+b_m^{\ 2})+\lambda_2(\| p_u \|^2+\| q_m \|^2)] \quad (7)$$

3.2 SVD++

协同过滤推荐技术实现的基础是推荐系统

中用户的历史行为信息，主要有两种：一是用户通过系统明确展示给我们的反馈信息（如评分等），称之为显式反馈（explicit feedback），二是那些用户在系统中操作的，但没有明确反馈意图的行为（如浏览等），称之为隐式反馈（implicit feedback）。

一般地，推荐系统对用户的显式反馈比较敏感，希望尽可能多的得到用户的显式行为。但在实际问题中，部分用户并不喜欢给予物品明确的评价，没有显式反馈意图，所以用户的显式信息比较难获取，而且在很大程度上存在缺失。虽然，用户没有留下大量的显式反馈数据，但系统对用户的操作行为存有记录，而这些被称为隐式反馈的行为记录蕴含了用户的观点或偏好，这对推荐系统分析用户特征也是非常有帮助的。因此，在Bias-SVD的基础上引入用户对物品的隐式反馈信息，对评分预测模型做进一步修正，可得SVD++模型[8]：

$$\hat{r}_{um} = b_{um} + q_m^T\left(p_u + \frac{1}{\sqrt{|N(u)|}}\sum_{j\in N(u)} y_j\right) \quad (8)$$

SVD++中将用户定义为 $p_u + \frac{1}{\sqrt{|N(u)|}}\sum_{j\in N(u)} y_j$，其中 P_u，是用户因子向量，可以通过显式评分数据学习得到，$\frac{1}{\sqrt{|N(u)|}}\sum_{j\in N(u)} y_j$ 是用户的隐式反馈部分，$N(u)$ 表示用户 u 的隐式反馈集合，Y_j 表示用户隐式评分下的物品因子向量。q_m 表示显式评分下的物品因子向量。通过最小化RMSE可建立目标函数：

$$C = \min \sum_{(u,m)\in T} (r_{um} - b_{um} - q_m^T(p_u + \frac{1}{\sqrt{|N(u)|}}\sum_{j\in N(u)} y_j))^2 + \lambda_1(b_{um}^2) + \lambda_2(\|p_u\|^2 + \|q_m\|^2) + \lambda_3\sum_{j\in N(u)}\|y_j\|^2] \quad (9)$$

SVD++补充考虑了隐式反馈行为对评分预测结果的影响，在原有隐语义模型的基础上，将用户因子分为显式评分和隐式评分两部分，综合考虑了影响评分预测结果的因素，使得模型信息更加完整。但SVD++增加了未知参数的数量，对模型的学习提出了更高要求。

3.3 Asymmetric-SVD

如果在SVD++模型的基础上，将每个物品 m 与三个维度相同的物品因子向量关联，即 p_m、y_m、$x_m \in R^f$。那么，用户因子向量没有提供显式参数，而是考虑用户评分过的物品，利用用户的历史评分偏好建立用户特征向量，则得到考虑邻域和隐式反馈的Asymmetric-SVD[8]模型：

$$\hat{r}_{um} = b_{um} + q_m^T\left(|R(u)|^{-\frac{1}{2}}\sum_{j\in R(u)}(r_{uj} - b_{uj})x_j + |N(u)|^{-\frac{1}{2}}\sum_{j\in N(u)} y_j\right) \quad (10)$$

$R(u)$ 表示用户 u 的显式评分集合，X_j 表示在给定的显式评分集中物品 j 的插值权重，其目标函数定义如公式（11）：

$$C = \min \sum_{(u,m)\in T} (r_{um} - \hat{r}_{um})^2 + \lambda_1(b_u^2 + b_m^2) + \lambda_2(\|p_u\|^2 + \|q_m\|^2) + \lambda_3\sum_{j\in R(u)}\|x_j\|^2 + \lambda_4\sum_{j\in N(u)}\|y_j\|^2 \quad (11)$$

该模型相比上述几种模型具有以下优点：

（1）参数少。一般个性化推荐系统中用户的数量远大于物品，用物品参数表示用户参数，减少了未知参数的个数，降低了模型复杂度。

（2）新用户处理。Asymmetric-SVD模型在处理推荐系统的冷启动问题上表现出特有的优势。因为该模型不需要生成用户参数，对于新用户而言，只要他们对系统提供了反馈，系统就可以及时产生推荐，而不需要重新训练模型或估计参数。并且，系统还可以根据用户的新行为，及时更新用户的偏好特征。但对于新物品而言，参数的学习还是需要通过训练得到的。Asymmetric-SVD模型体现了用户和物品的非对称性，这恰恰也符合推荐系统的实际应用。一方面，推荐系统对新用户推荐的及时性要求比较高，另一方面，系统中新物品的引入具有时段性，这给新物品参数的估计提供了重新训练的时间。

（3）解释性。隐语义模型，例如Base-SVD对评分预测结果的解释性是比较差的，因为它通过用户因子向量这个中间层表达用户偏好，将

可解释用户偏好的行为和预测结果分开了。而 Asymmetric-SVD 并没有对用户层面做抽象处理，直接通过用户的历史反馈行为预测结果，这可以帮助系统找出对预测结果影响最大的那些历史行为。

（4）整合隐式反馈。Asymmetric-SVD 模型综合考虑了用户的显式和隐式反馈信息，当用户的显式反馈数据较多，$N(u)$ 则比较大，显式反馈作用明显；当用户隐式反馈数据较多，则 $R(u)$ 比较大，隐式反馈作用明显。当然，单个显式反馈比单个隐式反馈价值更大，至于两者的比例，需要通过设置合适的参数 X_j、Y_j 和不断的学习进行设定。

4 实验分析

4.1 评价指标

4.1.1 平均绝对误差

平均绝对误差 MAE[9] 是推荐系统准确度评价中比较常用的一种，它计算用户实际评分与预测评分之间的平均绝对误差，具体计算如公式（12）所示

$$MAE=\frac{1}{|T|}\sum_{(u,m)\in T}|\hat{r}_{um}-r_{um}| \quad (12)$$

其中，T 代表测试集，[| T |] 即所有 user-item 评分对的总数，r_{um} 表示用户对物品的实际打分。$\hat{r}_{um}$ 表示预测评分，推荐结果的误差值是所有用户预测评分与真实打分误差的平均。

4.1.2 均方根误差

均方根误差与平均绝对误差相类似，都是准确度评价指标，自 Netflix Prize 竞赛之后受到广泛应用。均方根误差 RMSE[9] 对实际评分和预测评分之间的误差做平方处理，加重了因预测不准确而产生绝对误差的惩罚，对算法的要求更加严格。均方根误差的具体定义如公式（13）：

$$RMSE=\sqrt{\frac{\sum_{(u,m)\in P}(r_{um}-\hat{r}_{um})^2}{|T|}} \quad (13)$$

4.2 实验数据集

本文对隐语义模型的几种典型算法的评估实验主要基于 MovieLens 数据集 [10]。该数据集包括用户 ID 信息、物品 ID 信息，用户对物品的评分信息等，数据集的详细信息可见表 1。实验过程将评分数据集分为训练集（80%）和测试集（20%），共随机划分为 5 组，最终实验结果以 5 次交叉验证得到的 MAE 平均值和 RMSE 平均值为最终衡量标准。

表 1 MovieLens 数据集

数据集	用户数	电影数	评分数	训练集比例 x
MovieLens	943	1682	100000	80%

4.3 实验参数

对 Base-SVD、Bias-SVD、SVD++ 和 Asymmetric-SVD 这四个算法，分别通过实验讨论它们在显式评分预测问题中的性能特点。算法中涉及的重要参数主要包括：

（1）隐特征维度 F：隐语义模型的核心思想就是将评分矩阵进行降维，分解得到用户特征矩阵 P 和物品特征矩阵 Q，特征的 F 维度代表了保留评分矩阵信息的多少，F 越大，保留信息越多，但降维的优点就难以体现；F 越小，保留信息越少，但预测结果可能会出现较大偏差。因此，F 值的确定需要根据模型的特点、实际应用场景和实验情况等因素综合考虑来确定。对于受隐特征维度影响大的模型，需要慎重选择最佳的 F 值；而对于受隐特征维度影响小的模型，操作过程中可以减少对其选择问题的考虑。

（2）学习速率 α：学习速率的大小直接关系到模型迭代过程参数的改变量，如果 α 过大，模型收敛速度比较快，而且在梯度下降过程中很有

可能会跳过最优解；如果 α 过小，模型收敛所需的迭代次数就非常多，收敛速度比较缓慢。当然，对于不同模型，学习速率的取值是有差异的。为了避免出现跳过最优解的现象，可以在实验过程中采用变化的学习速率方法，比如学习速率的初始值可以选得精度大一些，每次迭代就缩小一点，如 $\alpha=\alpha*0.9$ 等。

（3）正则项系数 λ：λ 的选择也是需要根据算法本身、数据集等的特点，通过多次真实的实验来确定，如果选择过大，欠拟合问题就会出现；如果选择太小，正则化的效果就会不明显。

此外，在实验开始前，需要对未知参数即用户特征因子 P 和物品特征因子 Q 在每个隐特征维度上的因子变量赋予初始值，然后才能通过不断迭代沿着梯度方向往下寻找模型最优解。这里，本文在实验中采用的方法是随机产生 0 ~ 1 的浮点数，并考虑隐特征维度 F，计算 $0.1\times rand(0,1)/sqrt(F)$ 作为 P 和 Q 因子的初始值，同时以多次实验结果的平均值作为最终预测结果。

4.4 结果分析

（1）隐特征维度 F 对算法性能的影响

隐语义模型在实现过程中，隐特征维度的选择是一个难点，它关系到保留用户和物品评分信息多少的问题。为了探讨隐特征个数 F 在隐语义模型中的作用，以及对最终预测结果的影响，实验设置相同的学习速率 α 和正则项系数 λ，即 $\alpha=0.005\times0.9$、$\lambda_1=0.05$、$\lambda_2=0.1$、$\lambda_3=0.0175$，作为实验参数，在训练集上迭代 100 次，改变 F 的值，经多次反复实验得到的预测结果作为分析对象，讨论几种算法受隐特征维度 F 的影响情况。经五次交叉验证得到不同隐语义模型的预测误差 RMSE 和 MAE 分布情况，部分数值见表 2，分布曲线如图 2、图 3 所示。

表 2 不同隐语义模型在隐特征维度 F 变化下的预测误差分布（部分）

模型	Base-SVD		Bias-SVD		SVD++		Asymmetric-SVD	
F	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
20	0.995845	0.786098	0.947078	0.750418	0.941204	0.745268	0.923481	0.730193
40	0.995846	0.786102	0.94708	0.750421	0.94117	0.74524	0.9225	0.728955
60	0.995844	0.786096	0.947078	0.750418	0.941192	0.745255	0.922501	0.728906
80	0.995843	0.7861	0.947079	0.750419	0.94119	0.745257	0.922074	0.728672
100	0.995841	0.786098	0.947079	0.750419	0.941195	0.74526	0.922016	0.728671
150	0.99584	0.786096	0.947079	0.750419	0.941185	0.745251	0.921809	0.728443
200	0.995837	0.786096	0.947079	0.750419	0.941187	0.745252	0.921856	0.72851
300	0.995833	0.786095	0.947079	0.750419	0.941187	0.745239	0.921705	0.728393
500	0.995839	0.786097	0.947075	0.750413	0.941189	0.745241	0.92168	0.728359
700	0.99609	0.786283	0.947076	0.750413	0.941183	0.745234	0.921743	0.728329
1000	0.995839	0.786097	0.947079	0.750419	0.941213	0.745264	0.921738	0.728311

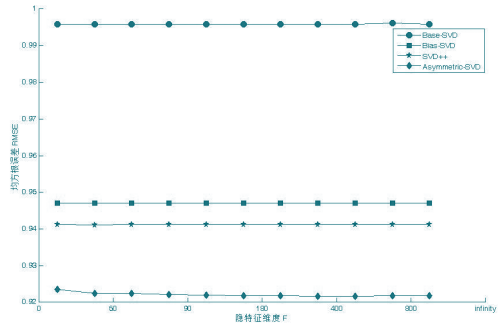

图 2　不同隐语义模型随隐特征维度 *F* 变化的 RMSE 值分布曲线

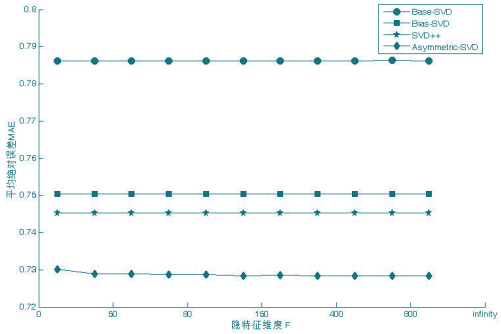

图 3 不同隐语义模型随隐特征维度 *F* 变化的 MAE 值分布曲线

由表 2 和图 2、图 3 可知，模型 Base−SVD、Bias−SVD 和 SVD++ 随隐特征维度 *F* 的不断变化，均方根误差 RMSE 和平均绝对误差 MAE 值变化非常小，尤其是模型 Bias−SVD，它在不同 *F* 值取得的预测误差，变化的数量级约为 10^{-5} 左右。同时，从分布曲线的波动幅度来看，这三种模型几乎接近一条水平直线，保持相对平稳的状态，这表明隐特征维度 *F* 对以上三种模型的推荐效果或预测精度影响并不大。因此，在对这三种模型进行实际操作过程中，可以减少对隐特征维度选择问题的考虑。

然而，模型 Asymmetric−SVD，在隐特征维度 *F* 较小时，预测误差下降的速度比较快，随着 *F* 的增加，下降速度逐渐变得缓和。表明该模型在隐特征维度 *F* 偏小时，取得的预测误差较大；随着隐特征维度的增加，预测误差先逐渐减小，然后当 *F* 达到一定维度之后，预测误差就不再减小，趋于稳定。因此，可以认为模型 Asymmetric−SVD 受隐特征维度 *F* 的影响相比其他模型要大，推荐精度随 *F* 的变化要明显。由此可知，在实际推荐系统中，对于不同的隐语义模型，隐特征个数的选择应有所区别，对于受影响大的模型，应经过多次反复实验，确定最终维度；对于受影响小的模型，可以减少对隐特征维度选择的考虑。

此外，观察图 2 和图 3，模型 Base−SVD 预测效果最差，RMSE 和 MAE 值最大，变化曲线处于最高位置；其次是引入偏置项的隐语义模型 Bias−SVD；然后是考虑隐反馈的模型 SVD++；而融合邻域的模型 Asymmetric−SVD 预测效果最佳，分布曲线位于最低位置。这也验证了在本次实验中，保持相同参数设置的条件下，改进的模型 Asymmetric−SVD 能够提高隐语义模型的预测精度。同时也从侧面反映了随着模型的不断改进，考虑的因素不断增加，预测误差逐渐减小，推荐效果不断提高。当然，也会存在一些缺陷，如模型复杂度的提高，需要消耗更大的时间和空间等，这在实际推荐系统设计中是需要考虑的重要因素。

（2）学习速率 α 对算法性能的影响

学习速率控制的是算法在迭代过程中参数的改变量，它的大小直接关系到模型最后的预测结果。本次实验采用固定隐特征维度 F=40，正则项系数不变，改变学习速率 α 和相应的迭代次数，对不同模型的预测误差变化情况进行分析，四种算法的实验结果分别如图 4 至图 7 所示。

从学习速率和迭代次数的变化关系来看，以模型 Asymmetric−SVD 为例，观察图 7 发现，学习速率由 0.007 降到 0.001，模型的预测误差 RMSE 和 MAE 的曲线变化有所不同。例如，当 α=0.007，迭代次数越小，模型预测误差越低，推

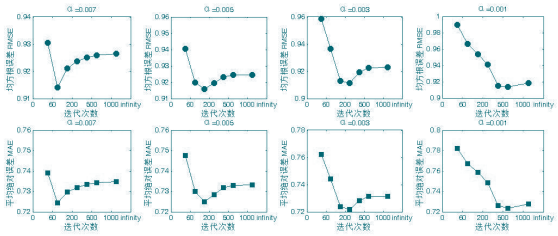

图 4 模型 Base-SVD 学习速率和迭代次数变化曲线

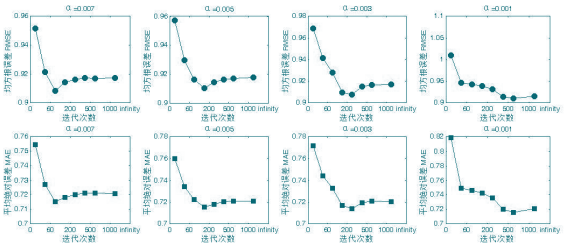

图 5 模型 Bias-SVD 学习速率和迭代次数变化曲线

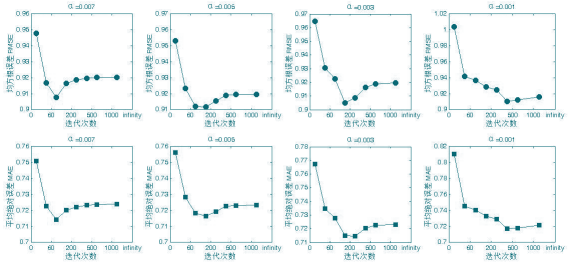

图 6 模型 SVD++ 学习速率和迭代次数变化曲线

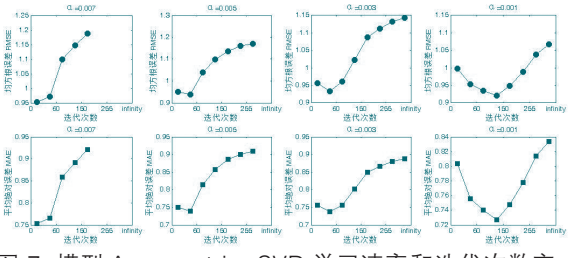

图 7 模型 Asymmetric-SVD 学习速率和迭代次数变化曲线

荐效果越好，迭代次数变大，预测误差反而增大，甚至得到趋于无穷大的数值；当 α=0.005 时，预测误差随迭代次数的增加先有所下降，取得较低的预测误差，但随后迭代次数再变大，预测误差又随之升高；当 α 继续下降，模型取得较低预测误差要求的迭代次数越多，而得到无穷大数值的情况也越不容易出现。

由于论文中采用的是有限次数的实验，其他模型学习速率和迭代次数的变化曲线特征可能没有模型 Asymmetric-SVD 明显，但都有类似的变化趋势，只是不同模型对学习速率的具体数值要求有所不同而已。如图 4，模型 Base-SVD，取得较低误差的迭代次数相对比较大，而且在学习速率 α 为 0.007 至 0.001 之间，有限次数的实验过程中，预测误差也并未出现无穷值。但随迭代次数的不断增加，预测误差呈现上升的趋势。假设，当迭代次数无限大时，预测误差可能也会取得无穷值，跳过最优解。

从学习速率对模型性能影响的角度来看，学习速率由 0.007 下降到 0.001，不同模型取得的局部最低预测误差值，以及误差的变化情况都有所不同。从表 3 的学习速率和迭代次数变化情况来看，学习速率逐渐降低，预测误差也逐渐减小，但相应迭代次数却不断增加，这也进一步验证了更小的学习速率能够取得较好的预测误差，但对算法效率的要求比较大。此外，四种模型在学习速率不断改变的情况下，取得的预测误差值具体变化程度有所不同。

如模型 Bias-SVD 变化最小，它的 RMSE 值从 0.90827 降到 0.90688，减少了 0.00139，MAE 值从 0.71526 降到 0.71329，减少了 0.00197；模型 Asymmetric-SVD 变化最大，它的 RMSE 值从 0.95203 降到 0.92134，减少了 0.03069，MAE 值从 0.75338 降到 0.72725，减少了 0.02613。由此可知，模型 Asymmetric-SVD 受学习速率影响的程度最大，模型 Bias-SVD 最小。而且，在实验过程中发现，相同学习速率下，模型 Asymmetric-SVD 相比其他模型更容易跳过最优解。因此，对该模型在实际操作中应选择较小的学习速率。

实验结果表明：当模型的学习速率取值偏大，取得较好推荐效果所需的迭代次数比较少，模型的收敛速度比较快，若迭代次数过大，预测误差容易趋于无穷大，很有可能在沿着梯度方向寻找局部最优解的时候会跨过最优解；当学习速率取

值偏小，取得较好推荐效果所需的迭代次数比较多，模型的收敛速度比较缓慢，需要很长时间才能找到局部最优解。虽然，更小的学习速率可以产生更低的预测误差，但同时算法的收敛速度也变得更慢。因此，在实际算法应用中，除了考虑模型本身的特性，还应根据推荐系统的内存空间、时间性能要求等因素，综合考虑以选择较好的学习速率和迭代次数等。

结果表明：隐特征维度对 Base-SVD、Bias-SVD 和 SVD++ 影响较小，对模型 Asymmetric-SVD 影响较大，且该模型的推荐结果预测精度较其他模型要好。由此证明，融合邻域和考虑隐反馈的隐语义模型能够取得较好的推荐效果。此外，针对控制算法参数改变量的学习速率，从迭代次数和对模型性能影响的角度，分析和讨论了它在算法实现中存在的问题和作用，得出该参数的选择对模型的预测结果影响较大，最终参数值的确定需综合考虑算法本身和推荐系统的需求。

表 3 不同学习速率和迭代次数下模型局部最低预测误差分布

Base-SVD				Bias-SVD			
学习速率	迭代次数	RMSE	MAE	学习速率	迭代次数	RMSE	MAE
0.007	95	0.91409	0.72437	0.007	95	0.90827	0.71526
0.005	195	0.91572	0.72485	0.005	195	0.91038	0.71564
0.003	255	0.91157	0.72165	0.003	255	0.90792	0.7143
0.001	700	0.91022	0.72056	0.001	700	0.90688	0.71329
SVD++				Asymmetric-SVD			
学习速率	迭代次数	RMSE	MAE	学习速率	迭代次数	RMSE	MAE
0.007	95	0.91022	0.72056	0.007	5	0.95203	0.75338
0.005	195	0.91022	0.72056	0.005	55	0.93763	0.7391
0.003	195	0.91022	0.72056	0.003	55	0.93313	0.7383
0.001	500	0.91022	0.72056	0.001	195	0.92134	0.72725

注：实验过程存在一定的局限性，比如有限次数的实验、参数的设定等问题，所以实验得到的预测误差值并不能完全反映模型的最优解，只能作为局部问题讨论的参考数据，深入的分析需要更多反复的实验。

5 结论

本文重点介绍了隐语义模型的基本原理、学习算法和评价指标等，分别对 Base-SVD、Bias-SVD、SVD++ 和 Asymmetric-SVD 等四种典型算法的核心思想、模型表示和学习方法做了详细描述，并通过实验对不同算法的性能进行比较分析。

参考文献

[1] Koren Y. Factor in the Neighbors: Scalable and Accurate Collaborative Filtering[J]. ACM Transactions on Knowledge Discovery from Data

(TKDD), 2010, 4(1):1-17.

[2] Luo X, Ouyang Y, Zhang X. Improving Latent Factor Model Based Collaborative Filtering via Integrated Folksonomy Factors[J]. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2011, 19(2):307-327.

[3] Hofmann T. Latent Semantic Models for Collaborative Filtering[J]. ACM Transactions on Information Systems, 2004, 22(1):89-115.

[4] Wang Q, Xu J, Li H, et al. Regularized latent semantic indexing[C]// Proceeding of the, International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July. 2011:685-694.

[5] Blei D M, Ng A Y, Jordan M I. Latent Dirichlet Allocation[J]. Journal of Machine Learning Research, 2003(3):993-1022.

[6] Koren Y, Bell R, Volinsky C. Matrix Factorization Techniques for Recommender Systems[J]. Computer, 2009, 42(8):30-37.

[7] Koren Y. The BellKor Solution to the Netflix Grand Prize[J]. Netflix Prize Documentation, 2009:1-10.

[8] Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C]// ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August. 2008:426-434.

[9] Adomavicius G, Tuzhilin A. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions[J]. IEEE Transactions on Knowledge & Data Engineering, 2005, 17(6):734-749.

[10] GroupLens. Moviewlens 100K Dataset. [EB/OL]. [2016-03-10].http://grouplens.org/datasets/movielens/100k/.