Journal of Computer Applications
计算机应用,2015,35(5):1328 - 1332,1341
ISSN 1001-9081
CODEN JYIIDU
2015-05-10
http://www.joca.cn

文章编号:1001-9081(2015)05-1328-05
doi:10.11772/j.issn.1001-9081.2015.05.1328

融合显/隐式反馈的协同排序算法

李　改[1,2,3*]

(1.顺德职业技术学院 电子与信息工程学院,广东 顺德 528333;　2.中山大学 信息科学与技术学院,广州 510006;
3.中山大学 软件研究所,广州 510275)

(*通信作者电子邮箱 ligai999@126.com)

摘　要:之前有关协同排序算法的研究没有充分利用数据集中信息的问题,要么只侧重于研究显式评分数据,要么只侧重于研究隐式评分数据,目前还没有人运用排序学习的思想把二者结合起来进行研究。针对之前研究的不足,在最新的扩展的少即是好协同过滤(xCLiMF)模型和最经典的变形的奇异值分解(SVD++)算法的基础上,提出了一种融合显/隐式反馈的协同排序算法MERR_SVD++来直接优化排序学习的评价指标ERR。在实际数据集上实验验证,与经典的xCLiMF、Cofi排序(CofiRank)、PopRec、Random算法相比,MERR_SVD++算法在归一化折损累积增益(NDCG)和预期的相关性排序(ERR)这两个评价指标下性能均提高了25.9%以上,而且算法运算时间与评分点个数线性相关。由于MERR_SVD++算法推荐精度高、可扩展性好,因此适用于处理大数据,在互联网信息推荐领域具有广泛的应用前景。

关键词:推荐系统;协同过滤;协同排序;隐式反馈;显式反馈

中图分类号:TP3　　**文献标志码**:A

Collaborative ranking algorithm by explicit and implicit feedback fusion

LI Gai[1,2,3*]

(1. *School of Electronics and Information Engineering, Shunde Polytechnic, Shunde Guangdong* 528333, *China*;
2. *School of Information Science and Technology, Sun Yat-Sen University, Guangzhou Guangdong* 510006, *China*;
3. *Software Institute, Sun Yat-Sen University, Guangzhou Guangdong* 510275, *China*)

Abstract: The problem of the previous research about collaborative ranking is that it does not make full use of the information in the dataset, either focusing on explicit feedback data, or focusing on implicit feedback data. Until now, nobody researches collaborative ranking algorithm by explicit and implicit feedback fusion. In order to overcome the defects of prior research, a new collaborative ranking algorithm by explicit and implicit feedback fusion named MERR_SVD++ was proposed to optimize Expected Reciprocal Rank (ERR) based on the newest Extended Collaborative Less-is-More Filtering (xCLiMF) model and Singular Value Decomposition++ (SVD++) algorithm. The experimental results on practical datasets show that, the values of Normalized Discounted Cumulative Gain (NDCG) and ERR for MERR_SVD++ are increased by 25.9% compared with xCLiMF, Cofi Ranking (CofiRank), PopRec and Random collaborative ranking algorithms, and the running time of MERR_SVD++ showed a linear correlation with the number of ratings. Because of the high precision and the good expansibility, MERR_SVD++ is suitable for processing big data, and has wide application prospect in the field of Internet information recommendation.

Key words: recommended system; collaborative filtering; collaborative ranking; implicit feedback; explicit feedback

0　引言

互联网的不断发展以及电子商务的蓬勃增长使得互联网上的信息总量爆炸性增长,导致“信息过载”,也就是尽管互联网上的信息众多,但互联网中的信息量大大高于网民所能获取、承受或需要的信息量,大量无关的、没用的、冗余的信息严重干扰了网民对相关有用信息的准确分析和正确选择。如何高效、快捷地获取所需要的个性化信息成为广大网民的迫切需求。在此背景下推荐系统应运而生,其中运用最广泛的是基于协同过滤的推荐算法[1-3]。

近来协同过滤算法在学术界和工业界得到了广泛研究。协同过滤算法所处理的数据类型主要分为两类:一类是具有明确偏好的显式数据[1,3],如评分(0~5分);另一类则是不明确的隐式数据[2],如对网页点击与否。按所使用的机器学习方法的类型协同过滤算法也主要分为两类:一类是基于评分预测的协同过滤算法(Collaborative Filtering, CF)[1-3];另一类是基于排序学习思想的协同排序算法(Collaborative Ranking, CR)[4-6]。基于评分预测的协同过滤算法主要通过矩阵分解模型来预测用户对缺失推荐对象的评分值,然后以

收稿日期:2014-12-19;**修回日期**:2015-02-03。　**基金项目**:国家自然科学基金资助项目(61003140,61033010);2011年度高校基本科研业务费中山大学青年教师培育项目(理工科)(111gpy58);佛山市产学研专项资金项目(2012HC100303);广东省高等职业教育教学改革2014年度项目(201401294);广东省高等职业教育教学改革2013年度项目(20130201109);广东省高等职业教育教学改革2012年度项目(20120302050);顺德职业技术学院2013年度校级教学改革与研究重点项目(2013-SZJGXM13)。

作者简介:李改(1981-),男,湖北松滋人,讲师,博士研究生,CCF会员,主要研究方向:机器学习、人工智能、推荐系统。

用户对缺失推荐对象的评分值来对缺失推荐对象进行排序,再把排好序的缺失推荐对象推荐给用户。协同排序的核心思想是对每个用户直接产生排好序的推荐对象序列,而非对每个推荐对象预测评分值后再排序。这个思想的合理性在于:实际的推荐系统中用户关心的是排好序的推荐对象序列,而非每个推荐对象的评分预测值。

之前对基于排序学习思想的协同过滤算法的研究要么只针对显式评分数据,要么只针对隐式评分数据;而在实际的应用系统中,往往既包含显式评分数据,又包含隐式评分数据。变形的奇异值分解(Singular Value Decomposition ++, SVD ++)算法[7]是目前为止最经典的融合显/隐式反馈的协同过滤算法,但其是基于评分预测的协同过滤算法。当前基于排序学习思想的协同排序算法的一大研究热点是通过直接优化排序学习的评价指标来提高协同排序算法的性能。

为了进一步提高协同排序算法的性能,本文在最新的扩展的少即是好协同过滤(Extended Collaborative Less-is-More Filtering, xCLiMF)模型和最经典的SVD ++ 算法的基础上,提出了一种融合显/隐式反馈的协同排序算法MERR_SVD ++,该算法通过同时利用显/隐式反馈信息来直接优化排序学习的评价指标:预期的相关性排序(Expected Reciprocal Rank, ERR)[8]。在实际数据集上实验验证,本文算法在各个评价指标下均优于之前的协同排序算法。

1　相关工作

当前通过直接优化排序学习评价指标的协同排序算法主要分为两类:一类是直接优化排序学习的评价指标的单类协同过滤(One Class Collaborative Filtering, OCCF)算法;另一类是面向显式评分(rating)数据的直接优化排序学习的评价指标的协同过滤算法。近来,运用基于排序学习的思想来研究OCCF 问题的算法中最有名的是 Rendle 等[4]提出运用贝叶斯个性化排序(Bayesian Personal Ranking, BPR)模型,该模型通过直接优化排序学习的评价指标——接收器操作特性曲线下的面积(Area Under the receiver operating characteristic Curve, AUC)来从排序学习的角度研究 OCCF 问题。除此之外,Pan 等[9]提出了基于组的贝叶斯个性化排序(Group preference based Bayesian Personal Ranking, GBPR)模型,GBPR 模型是在 BPR 模型基础上考虑了用户的邻居对该用户偏好的影响,也是直接优化评价指标 AUC 来提高协同排序算法的性能。Shi 等[10]提出了少即是好协同过滤(Collaborative Less-is-More Filtering, CLiMF)模型,CLiMF 模型通过直接优化评价指标——平均相关性排序(Mean Reciprocal Rank, MRR)对推荐对象进行排序。

目前面向显式评分数据的直接优化排序学习的评价指标协同过滤算法也出现了一些,如 Weimer 等人通过改进最大边界矩阵分解模型[11],提出了一种优化评价指标——归一化折损累积增益(Normalized Discounted Cumulative Gain, NDCG)的协同排序算法——Cofi 排序(Cofi Ranking, CofiRank)[12]。Shi 等[8]提出了 xCLiMF 模型,xCLiMF 模型通过直接优化评价指标 ERR 来对推荐对象进行排序,是 CLiMF 模型的扩展版。Liu 等[13-14]提出了两种基于成对损失函数的协同排序算法——概率潜在偏好分析(probabilistic Latent Preference Analysis, pLPA)[13]和 Eigen 排序(Eigen Ranking, EigenRank)算法[14],pLPA 和 EigenRank 不是直接优化评价指标的协同排序算法,而是通过改进基于概率潜在语义分析的协同过滤算法和基于最近邻的协同过滤算法而提出的基于排序学习思想的协同过滤算法。

2　基本定义和SVD ++ 算法简介

2.1　基本定义

在本文中矩阵用斜体大写字母表示(如:$\boldsymbol{X}$),标量用小写字母表示(如:i, j)。给定一个矩阵 $\boldsymbol{X}$,X_{ij} 表示它的一个元素,$\boldsymbol{X}_i$ 表示矩阵 $\boldsymbol{X}$ 的第 i 行,$\boldsymbol{X}_j$ 表示矩阵 $\boldsymbol{X}$ 的第 j 列,$\boldsymbol{X}^{\mathrm{T}}$ 表示矩阵 $\boldsymbol{X}$ 的转置。

2.2　SVD ++ 算法简介

给定一个矩阵 $\boldsymbol{X} = (X_{ij})_{m\times n}$($m$ 表示用户数,n 表示推荐对象数),$\boldsymbol{X}$ 表示显式评分矩阵,其中 $X_{ij} \in \{0,1,2,3,4,5\}$ 或者未知。在这里希望找到一个低秩矩阵 $\hat{\boldsymbol{X}} = (\hat{X}_{ij})_{m\times n}$ 来逼近矩阵 $\boldsymbol{X}$,其中 $\hat{\boldsymbol{X}} = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{V}, \boldsymbol{U} \in \boldsymbol{C}^{d\times m}, \boldsymbol{V} \in \boldsymbol{C}^{d\times n}$,$\boldsymbol{U}$、$\boldsymbol{V}$ 表示用户和推荐对象的显式特征矩阵,d 表示特征个数,一般 $d \ll k$,k 表示矩阵 $\boldsymbol{R}$ 的秩,$k \leqslant \min(m,n)$。

如果 $\boldsymbol{X}$ 表示隐式评分矩阵,则 $X_{ij} \in \{0,1\}$ 或者未知。在这里也希望找到一个低秩矩阵 $\hat{\boldsymbol{X}} = (\hat{X}_{ij})_{m\times n}$ 来逼近矩阵 $\boldsymbol{X}$,其中 $\hat{\boldsymbol{X}} = \boldsymbol{H}^{\mathrm{T}}\boldsymbol{W}, \boldsymbol{H} \in \boldsymbol{C}^{d\times m}, \boldsymbol{W} \in \boldsymbol{C}^{d\times n}$,$\boldsymbol{H}$、$\boldsymbol{W}$ 表示用户和推荐对象的隐式特征矩阵。

SVD ++ 算法是一种融合显/隐式反馈的基于评分预测和矩阵分解的协同过滤算法[7]。在SVD ++ 算法中,推荐对象的显式特征矩阵和隐式特征矩阵用同一个矩阵 $\boldsymbol{V}$ 表示。用户的特征矩阵表示为:

$$\boldsymbol{U}_u + |N(u)|^{-\frac{1}{2}} \sum_{j\in N(u)} \boldsymbol{W}_j \tag{1}$$

式(1)中:$\boldsymbol{U}_u$ 用来刻画用户显式反馈的打分,$|N(u)|^{-\frac{1}{2}} \sum_{j\in N(u)} \boldsymbol{W}_j$ 用来刻画用户隐式反馈的打分,其中 $N(u)$ 表示用户 u 给予了隐式反馈的推荐对象集合,$\boldsymbol{W}_j$ 表示推荐对象的隐式特征向量。

则SVD ++ 算法中 $\hat{X}_{ui}$ 的预测公式如下:

$$\hat{X}_{ui} = \left(\boldsymbol{U}_u + |N(u)|^{-\frac{1}{2}} \sum_{j\in N(u)} \boldsymbol{W}_j\right)^{\mathrm{T}} \boldsymbol{V}_i \tag{2}$$

3　融合显/隐式反馈的协同排序算法

3.1　融合显/隐式反馈的协同排序算法简介

在实际的应用中,用户通常是从前往后查看反馈回来的推荐结果,直到找到满意的推荐对象,用户查阅第 i 个位置的推荐对象的概率与用户对第 i 个位置之前的推荐对象的满意程度有关。信息检索领域重要的排序学习的评价指标——相关性排序(Reciprocal Rank, RR)[10]正是用来评价这样一种推荐对象之间相关度的评价指标。评价指标 ERR 是 RR 的改进版,用于评价拥有显式评分数据的推荐对象序列。Shi 等[8]最先把优化评价指标 ERR 用于显式反馈的协同排序。

如文献[8]中所示,用于评价对用户 u 所产生的推荐序列的 ERR_u 公式如下所示:

$$ERR_u = \sum_{i=1}^{n} \frac{r_{ui}}{R_{ui}} \prod_{j=1}^{n} (1 - r_{uj} I(R_{uj} < R_{ui})) \tag{3}$$

其中:$I(x)$ 是一个指示函数,如果 x 为真,则 $I(x)$ 的值为 1;

否则为 0。R_{ui} 表示对用户 u 所产生的推荐序列中推荐对象 i 所处的位次。r_{ui} 表示用户 u 对推荐对象 i 的满意概率，本文和文献[8]中一样设置为：

$$r_{ui} = \begin{cases} \dfrac{2^{X_{ui}} - 1}{2^{X_{\max}}}, & I_{ui} > 0 \\ 0, & I_{ui} = 0 \end{cases}$$

其中：I_{ui} 是一个指示函数，如果用户 u 对推荐对象 i 给过评分则 $I_{ui} > 0$，否则 $I_{ui} = 0$；X_{ui} 表示用户 u 给推荐对象 i 的评分值；$X_{\max}$ 表示用户 u 给过的最高评分值。

定义 $R(u)$ 为用户 u 赋予了显式反馈的推荐对象的集合，在用户仅仅给予了显式评分的数据集上 $N(u) = R(u)$，此时的隐式评分的数据集是把所有的显式评分均设置为 1 后所得到的数据集。如果数据集中既包含显式评分数据，又包含隐式评分数据，则 $N(u) = R(u) + M(u)$，其中 $M(u)$ 表示用户 u 仅仅给予了隐式反馈的推荐对象的集合，即用户 u 没有对 $M(u)$ 中的任何推荐对象赋予过显式评分。引入传统的融合显/隐式反馈的基于评分预测和矩阵分解的SVD ++ 算法，此时 $\hat{X}_{ui}$ 的预测公式如下：

$$\hat{X}_{ui} = \left(\boldsymbol{U}_u + |N(u)|^{-\frac{1}{2}} \sum_{j \in N(u)} \boldsymbol{W}_j\right)^{\mathrm{T}} \boldsymbol{V}_i$$

式(3)中的 R_{ui} 的取值由 $\hat{X}_{ui}$ 决定。如果在算法对用户 u 所产生的推荐序列的所有推荐对象的预测值中，$\hat{X}_{ui}$ 的值排在第 2 位，则 R_{ui} 的值为 2。至此，通过引入SVD ++ 算法，Shi 等提出的直接优化评价指标 ERR 来解决显式反馈的协同排序问题的方法也可以运用到融合了显/隐式反馈的数据集中。

和直接优化评价指标 RR 一样，在引入特征矩阵 $\boldsymbol{U}$、$\boldsymbol{V}$、$\boldsymbol{W}$ 后优化评价指标 ERR 的目标函数(3)变得不光滑、无法运用传统的优化技术来直接优化评价指标 ERR。为了解决这个优化问题，采用文献[10]中的方法，引入如下近似公式：

$$I(R_{uj} < R_{ui}) \approx g(\hat{X}_{uj} - \hat{X}_{ui}) \tag{4}$$

$$1/R_{ui} = g(\hat{X}_{ui}) \tag{5}$$

其中 $g(x)$ 是一个 logistic 函数，$g(x) = 1/(1 + \mathrm{e}^{-x})$。

把式(4)、(5)代入式(3)，得到如下光滑的近似的 ERR_u：

$$ERR_u = \sum_{i=1}^{n} r_{ui} g(\hat{X}_{ui}) \prod_{j=1}^{n} (1 - r_{uj} g(\hat{X}_{u(j-i)})) \tag{6}$$

其中 $\hat{X}_{u(j-i)} = \hat{X}_{uj} - \hat{X}_{ui}$。

运用 Jensen 不等式和对数函数的单调性，直接优化 ERR_u 和优化 $\ln\left(\frac{1}{|N(u)|} ERR_u\right)$ 效果一样[15]。

$$\ln\left(\frac{1}{|N(u)|} ERR_u\right) =$$

$$\ln\left[\sum_{j=1}^{n} \frac{r_{ui} g(\hat{X}_{ui})}{|N(u)|} \prod_{j=1}^{n} (1 - r_{uj} g(\hat{X}_{u(j-i)}))\right] \geqslant$$

$$\frac{1}{|N(u)|} \sum_{j=1}^{n} r_{ui} \ln\left[g(\hat{X}_{ui}) \prod_{j=1}^{n} (1 - r_{uj} g(\hat{X}_{u(j-i)}))\right] =$$

$$\frac{1}{|N(u)|} \sum_{j=1}^{n} r_{ui} \left[\ln g(\hat{X}_{ui}) + \sum_{j=1}^{n} \ln(1 - r_{uj} g(\hat{X}_{u(j-i)}))\right] \tag{7}$$

则目标函数变为：

$$L(\boldsymbol{U}_u, \boldsymbol{V}, \boldsymbol{W}) =$$

$$\sum_{j=1}^{n} r_{ui} [\ln g(\hat{X}_{ui}) + \sum_{j=1}^{n} \ln (1 - r_{uj} g(\hat{X}_{u(j-i)}))] \tag{8}$$

引入正则化规范，最终目标函数变为：

$$L(\boldsymbol{U}_u, \boldsymbol{V}, \boldsymbol{W}) =$$

$$\sum_{j=1}^{n} r_{ui} [\ln g(\hat{X}_{ui}) + \sum_{j=1}^{n} \ln (1 - r_{uj} g(\hat{X}_{u(j-i)}))] -$$

$$\frac{\lambda}{2} (\|\boldsymbol{U}\|^2 + \|\boldsymbol{V}\|^2 + \|\boldsymbol{W}\|^2) \tag{9}$$

3.2 算法求解和形式化描述

对最终目标函数(9)使用梯度下降方法分别对 $\boldsymbol{U}$、$\boldsymbol{V}$、$\boldsymbol{W}$ 求偏导得到如下公式：

$$\frac{\partial L}{\partial \boldsymbol{U}_u} = \sum_{i=1}^{n} r_{ui} \left[g(-\hat{X}_{ui}) \boldsymbol{V}_i + \sum_{j=1}^{n} \frac{r_{uj} g'(\hat{X}_{uj} - \hat{X}_{ui})(\boldsymbol{V}_i - \boldsymbol{V}_j)}{1 - r_{uj} g(\hat{X}_{uj} - \hat{X}_{ui})}\right] - \lambda \boldsymbol{U}_u \tag{10}$$

$$\frac{\partial L}{\partial \boldsymbol{V}_i} = r_{ui} \left[g(-\hat{X}_{ui}) \boldsymbol{V}_i + \sum_{j=1}^{n} r_{uj} g'(\hat{X}_{uj} - \hat{X}_{ui}) (1/[1 - r_{uj} g(\hat{X}_{uj} - \hat{X}_{ui})] - 1/[1 - r_{uj} g(\hat{X}_{ui} - \hat{X}_{uj})])\right] \left(\boldsymbol{U}_u + |N(u)|^{-\frac{1}{2}} \sum_{k \in N(u)} \boldsymbol{W}_k\right) - \lambda \boldsymbol{V}_i \tag{11}$$

$$\frac{\partial L}{\partial \boldsymbol{W}_i} = r_{ui} \left[g(-\hat{X}_{ui}) \boldsymbol{V}_i + \sum_{j=1}^{n} \frac{r_{uj} g'(\hat{X}_{uj} - \hat{X}_{ui})(\boldsymbol{V}_i - \boldsymbol{V}_j)}{1 - r_{uj} g(\hat{X}_{uj} - \hat{X}_{ui})}\right] - \lambda \boldsymbol{W}_i \tag{12}$$

算法的形式化描述如算法 1 所示。

算法 1　融合显/隐式反馈的协同排序算法(MERR_SVD ++)。

输入　训练数据集 $\boldsymbol{X}$，学习率 α，正则化参数 λ，特征数 d；

输出　特征矩阵 $\boldsymbol{U}$、$\boldsymbol{V}$、$\boldsymbol{W}$。

```
for u = 1,2,…,m do
  % 索引用户 u 赋予过显式评分或隐式评分的所有物品；
  N(u) = {j | X_ij > 0, 0 ≤ j ≤ n};
end
初始化特征矩阵 U^(0)、V^(0)、W^(0)；
设置变量 t = 0;          % 变量 t 用于记录算法迭代的轮数;
While 未收敛 do
  for u = 1,2,…,m do
    % 更新 U_u;
    U_u^(t+1) = U_u^(t) + α ∂L/∂U_u^(t);
    For i ∈ N(u) do
    % 更新 V_i, W_i;
      V_i^(t+1) = V_i^(t) + α ∂L/∂V_i^(t);
      W_i^(t+1) = W_i^(t) + α ∂L/∂W_i^(t);
    End
  End
  t = t + 1;
End
U = U^(t), V = V^(t), W = W^(t);
返回特征矩阵 U、V、W
```

在本文的实验中，特征矩阵 $\boldsymbol{U}$、$\boldsymbol{V}$、$\boldsymbol{W}$ 初始化为满足均值为 0，方差为 0.01 的标准正态分布值：N(0,0.01)。

3.3 算法的时间复杂度分析

本文先分析算法 1 迭代 1 轮的复杂度。由于评分矩阵 $\boldsymbol{X}$ 是一个稀疏矩阵,求用户 u 的偏导的式(10)和式(12)的时间复杂度为 $O(d\hat{n}^2m + dm)$,求用户 u 的偏导的式(11)的时间复杂度为 $O(d\hat{n}^2m + d\hat{n}m)$,其中 $\hat{n}$ 表示所有用户的 $|N(u)|$ 值的平均数。在这里 $\hat{n}^2 \ll m$,因此式(10)的时间复杂度与用户数 m 成正比。定义评分矩阵 $\boldsymbol{X}$ 中的评分点个数为 s,则 $s = \hat{n}m$。因此算法 1 迭代 1 轮的时间复杂度为 $O(d\hat{n}s)$,通常 $\hat{n} \ll s$。假定算法迭代 t 轮后收敛,因此算法总的时间复杂度为 $O(td\hat{n}s)$。由于评分矩阵 $\boldsymbol{X}$ 高度稀疏,所以算法 MERR_SVD ++ 具有很好的可扩展性,可用于处理大数据。可扩展性的详细实验分析见 4.4.3 节。

4 实验结果及分析

4.1 实验数据集

在本文实验中一共使用了 2 个数据集。

MovieLens(ML1m)数据集[8]:该数据集是 GroupLens 工作小组公布的 MovieLens 电影推荐系统中用户对电影评分信息。这个数据总共包含了 6000 个用户对接近 3700 部电影的 1 000 209个打分,其中打分数据是用 1 到 5 分来表示的,分数越高表示用户对该电影越喜欢。这个数据集的稀疏度是 95.53%。

Netflix 比赛数据集[7]:该数据集是由美国 DVD 在线租赁商 Netflix 公司在 2006 年发起的一项竞赛时公布的数据集。这个数据集包含了 480 189 个用户对 17 770 部电影 100 000 000条打分记录。该数据集的稀疏度达到了 99.88%。在本实验中,没有使用该数据集的所有数据,而是随机挑选了 100 000 个用户对其中 10 000 部电影打分作为实验数据集,其中每个用户至少对 100 部电影有过打分。

4.2 实验设置

在不同数据集,本文分别计算在给出不同数量用户打分数据的情况下各个算法的表现。例如条件“Given 5”表示对于每个用户随机挑选 5 部他们打过分的电影作为训练集。对于 MovieLens 数据集给出条件:“Given 5”、“Given 10”和“Given 15”;而对于 Netflix 数据集给出:“Given 10”、“Given 20”和“Given 30”这 3 个条件。

随机挑选用户打过分的 5 部电影和用户没打分的1 000 部电影合在一起作为用户的测试集。对于每个数据集中给定每个条件,分别对每个算法在各个评价标准下进行了 5 次实验,实验的最终结果是这 5 次实验的平均得分。对于矩阵分解的方法,设置的特征变量个数 d 为 10。通过交叉确认实验,MERR_SVD ++ 算法的学习率 α 设置为 0.001、正则化参数 λ 设置为 0.001 时的性能最好。其他算法的参数均参照相应参考文献中的设置。

4.3 实验的评价指标

本实验的评价指标用到了 $NDCG$[8,12]、ERR[8]。这 2 个评价指标的定义如下:

$$DCG_u = \sum_{k=1}^{n} \frac{2^{pref(k)} - 1}{\log(k+1)}$$

$$NDCG_u = \frac{DCG_u}{IDCG_u}$$

如果测试集中的第 i 个项目是用户浏览过的项目,则 $pref(i) = 1$,否则 $pref(i) = 0$。$IDCG_u$ 表示理想状态下,即所有推荐项目均按用户 u 的喜欢程度排序时的 DCG_u 值。$NDCG$ 就是所有用户 $NDCG_u$ 值的平均值。

计算对用户 u 所产生的推荐序列的 ERR_u 值如式(13)所示,ERR 就是所有用户 ERR_u 值的平均值。

$$ERR_u = \sum_{i=1}^{n} \frac{r_{ui}}{R_{ui}} \prod_{j=1}^{n} (1 - r_{uj} I(R_{uj} < R_{ui})) \tag{13}$$

$NDCG@5$ 和 $ERR@5$ 就是给所有用户返回 5 个推荐对象时的 $NDCG$ 值和 ERR 值。

4.4 实验结果

在节把本文所提出的MERR_SVD ++ 算法与如下 4 个经典的协同排序算法作比较。

Random 算法　随机抽取对象推荐给每个用户。

PopRec 算法　即对每个用户的测试对象按该对象的热门度来排序。测试集中的某个对象在训练集中被给予评分的用户数越多,该对象排名越靠前。

CofiRank 算法　一种优化排序学习的评价指标 $NDCG$ 的协同排序算法,该算法处理的是显式反馈数据,该算法的具体描述见文献[12]。

xCLiMF 算法　一种优化排序学习的评价指标 ERR 的最新的协同排序算法,该算法处理的是显式反馈数据,该算法的具体描述见文献[8]。

本文所有算法均在 Linux 操作系统下用基于 Java 的 ECLIPSE 开发平台编写,计算机配置为 4 颗 Intel Core i7 处理器,8 GB 内存。

4.4.1 本文算法和几个经典协同排序算法的性能比较

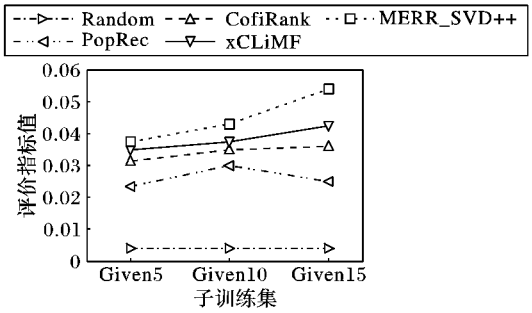
本文所提的MERR_SVD ++ 算法和几个经典协同排序算法的性能比较如图 1 所示。

从图 1 中可以看出,MERR_SVD ++ 算法在两个数据集的各种稀疏度的子数据集条件下,与经典的 xCLiMF、Cofi 排序(CofiRank)、PopRec、Random 算法相比在 $NDCG$ 和 ERR 这两个评价指标下性能均提高了 25.9% 以上,表明本文算法具有很强的抗数据稀疏性。在这里把本文所提MERR_SVD ++ 算法性能的提高归功于在融合显/隐式反馈信息的条件下直接优化评价标准 ERR 的结果。实验结果表明 ERR 指标的提高和 $NDCG$ 的提高是一致的,这说明通过直接优化 ERR 指标也能帮助提高测试数据的 $NDCG$ 得分。随着各个子训练集中用户打分数据的增加,各个算法的性能均有所提高,但 MERR_SVD ++ 算法相比其他算法的推荐性能提高更为显著,而且增加的幅度也正比于观测到的用户数据。这也进一步证明了本文所提算法的优越性。

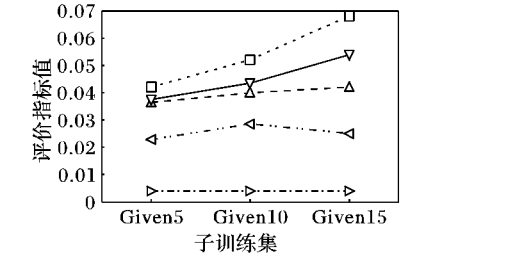
4.4.2 本文算法增加隐式反馈数据的实验结果

图 2 给出了在MERR_SVD ++ 算法中仅仅增加隐式反馈数据的效果图。也即此时 $N(u) = R(u) + M(u)$,其中 $M(u)$ 表示用户 u 仅仅赋予了隐式反馈的推荐对象的数量。其中纵轴表示评价指标的值;横轴表示在MERR_SVD ++ 算法中给各个用户增加的隐式反馈的数据量,即 $M(u)$ 的值。在本实验中对所有用户额外增加的隐式反馈数据的数量一样,使用的数据集是 Netflix 数据的子数据集 Given 10。图 2 显示,随着 MERR_SVD ++ 算法中隐式反馈数据的数量的增加,$NDCG@5$ 和 $ERR@5$ 这两个评价指标的值均同步近似线性增加。这说明,在MERR_SVD ++ 算法中如果只增加用户隐式反

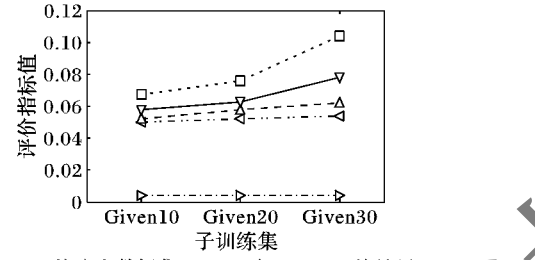
馈数据,实验效果也明显提高。这是本文算法的一大优点。



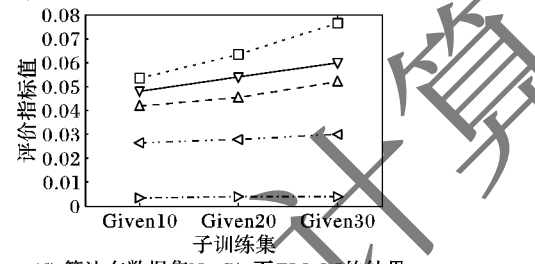
(a) 算法在数据集MovieLens下*NDCG*@5的结果



(b) 算法在数据集MovieLens下*ERR*@5的结果



(c) 算法在数据集Netflix下*NDCG*@5的结果



(d) 算法在数据集Netflix下*ERR*@5的结果

图 1 MERR_SVD ++ 算法与其他经典协同排序算法的比较

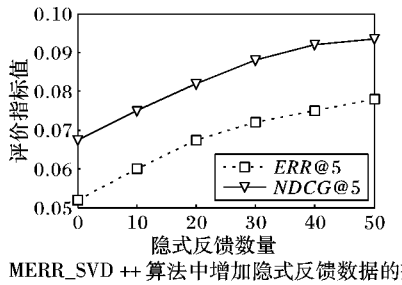


图 2 MERR_SVD ++ 算法中增加隐式反馈数据的效果

4.4.3 本文算法的可扩展性实验结果

图 3 给出了在MERR_SVD ++ 算法中仅仅增加训练集中用户比例对算法运行时间的影响。图中分别给出了在MovieLens 数据集的 3 个子数据集:“Given 5”、“Given 10”和“Given 15”上的实验结果。

从图 3 可以看出,随着训练集中用户占全部用户数的比例值的线性增加,在各个子数据集下算法MERR_SVD ++ 的训练时间也线性增加。即当固定用户的平均评分点的个数时,每一轮迭代时间就和用户数目接近线性关系,因此本文所提出的MERR_SVD ++ 算法的时间复杂度与数据集中可观测到的评分点的个数成线性关系。这也进一步证明了本文所提MERR_SVD ++ 算法的可扩展性。

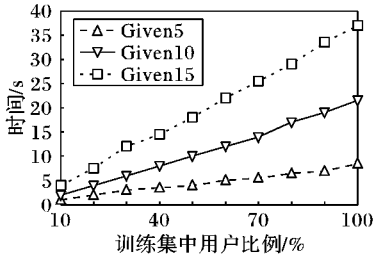


图 3 MERR_SVD ++ 算法中用户比例变化对算法运行时间的影响

5 结语

之前对协同排序算法的研究要么只侧重于研究显式评分数据,要么只侧重于研究隐式评分数据,目前还没有把显式评分数据和隐式评分数据结合起来研究的协同排序算法。为了克服之前研究的不足,本文提出了一种新的融合显/隐式反馈的协同排序算法。在 2 个实际数据集上实验验证,本文算法不仅在各个评价指标下均优于之前的协同排序算法,而且具有很好的可扩展性。

参考文献:

[1] 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 and Data Engineering, 2005, 17(6): 734 – 749.

[2] PAN R, ZHOU Y, CAO B, *et al*. One-class collaborative filtering [C]// Proceedings of the 22nd International Conference on Data Mining. Piscataway: IEEE, 2008: 502 – 511.

[3] HERNANDEZ-LOBOTA J M, HOULSBY N, GHAHRAMANI Z B. Probabilistic matrix factorization with non-random missing data[C]// Proceedings of the 31nd International Conference on Machine Learning. Piscataway: IEEE, 2014: 1257 – 1264.

[4] RENDLE S, FREUDENTHALER C, GANTNER Z, *et al*. BPR: Bayesian personalized ranking from implicit feedback[C]// Proceedings of the 22nd International Conference on Uncertainty in Artificial Intelligence. Montreal: AUAI Press, 2009: 52 – 461.

[5] LIU T Y. Learning to rank for information retrieval[M]. New York: Springer, 2011: 1 – 304.

[6] SUHRID B, SUMIT C. Collaborative ranking[C]// Proceedings of the 2012 ACM International Conference on Web Search and Data Mining. New York: ACM, 2012: 143 – 152.

[7] KOREN Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C]// Proceedings of the 25th International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2008: 426 – 434.

[8] SHI Y, KARATZOGLOU A, BALTRUNAS L, *et al*. xCLiMF: Optimizing expected reciprocal rank for data with multiple levels of relevance[C]// Proceedings of the 6th ACM Conference on Recommender Systems. New York: ACM, 2013: 431 – 433.

[9] PAN W, LI C. GBPR: group preference based Bayesian personalized ranking for one-class collaborative filtering[C]// Proceedings of the 26th International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2009: 667 – 676.

(下转第 1341 页)

从图 2 可看出,OAPSO 算法不仅寻优能力强,且收敛速度快,在单峰函数处理上优势非常明显,函数进化曲线几乎呈直线下降。观察图 3 可知,OAPSO 算法在处理复杂的多峰值函数时,在其他几种算法陷入局部最优、收敛速度变慢,甚至停滞不时,收敛速度依然具有相当强的优势,在 f_5、f_6 和 f_8 函数上,虽然 DCPSO 和 OAPSO 算法都能寻找到最优解,但 OAPSO 算法在大约 2 万次迭代次数时即寻找到最优解,而 DCPSO 确需要大概 4 万次迭代才能达到。因此,本文提出的 OAPSO 算法,通过设定的阈值将种群进化状态划分为正常状态和"早熟"状态,并让算法自适应选择不同的进化模式和学习策略,能很好地增强算法逃离局部最优能力和加速算法后期收敛速度。

5　结语

本文针对标准 PSO 算法进化模式单一、后期收敛速度慢、收敛精度低和易陷入"早熟"的缺点,提出一种具有反向学习和自适应逃逸功能的粒子群优化算法。算法通过阈值的设定来判断种群当前所处的进化状态,若算法处于"早熟"状态,选择添加了个体最优粒子反向解的速度进化模式和线性下降的惯性权重帮助粒子跳出局部最优,反之,算法采用固定的惯性权重和标准 PSO 算法的进化模式更新种群。仿真实验表明 OAPSO 算法比其他几种经典粒子群优化算法收敛精度更高、收敛速度更快。

参考文献:

[1] KENNEDY J, EBERHART R. Particle swarm optimization[C]// Proceedings of the 4th IEEE International Conference on Neural Networks. Piscataway: IEEE, 1995: 1942 - 1948.

[2] WANG L, YANG B, CHEN Y. Improving particle swarm optimization using multi-layer searching strategy[J]. Information Sciences, 2014, 274: 70 - 94.

[3] BEHESHTI Z, SHAMSUDDIN S M, HASAN S. MPSO: Median-oriented particle swarm optimization [J]. Applied Mathematics and Computation, 2013, 219(11): 5817 - 5836.

[4] KIRAN M S, GUNDUZ M. A recombination-based hybridization of particle swarm optimization and artificial bee colony algorithm for continuous optimization problems[J]. Applied Soft Computing, 2013, 13(4): 2188 - 2203.

[5] CHEN W, ZHANG J, LIN Y, *et al*. Particle swarm optimization with an aging leader and challengers[J]. IEEE Transactions on Evolutionary Computation, 2013, 17(2): 241 - 258.

[6] GUO T, LAN J, LI Y, *et al*. Adaptive fractional-order Darwinian particle swarm optimization algorithm[J]. Journal on Communications, 2014, 35(4): 130 - 140. (郭通, 兰巨龙, 李玉峰, 等. 自适应的分数阶达尔文粒子群优化算法[J]. 通信学报, 2014, 35(4): 130 - 140.)

[7] BEHESHTI Z, SHAMSUDDIN S M. CAPSO: Centripetal accelerated particle swarm optimization[J]. Information Sciences, 2014, 258: 54 - 79.

[8] SHI X, SUN H, LI J, *et al*. Particle swarm optimization algorithm with fast convergence and adaptive escape[J]. Journal of Computer Applications, 2013, 33(5): 1308 - 1312. (史小露, 孙辉, 李俊, 等. 具有快速收敛和自适应逃逸功能的粒子群优化算法[J]. 计算机应用, 2013, 33(5): 1308 - 1312.)

[9] TIZHOOSH H R. Opposition-based learning: a new scheme for machine intelligence[C]// Proceedings of the 2005 International Computational Intelligence for Modeling Control and Automation. Piscataway: IEEE, 2005: 695 - 701.

[10] MENDES R, KENNEDY J, NEVES J. The fully informed particle swarm: simpler, maybe better [J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 204 - 210.

[11] RATNAWEERA A, HALGAMUGE S, WATSON H. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients [J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 240 - 255.

[12] LIANG J J, QIN A K, SUGANTHAN P N, *et al*. Comprehensive learning particle swarm optimizer for global optimization of multimodal function[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(3): 281 - 295.

[13] ZHAN Z, ZHANG J, LI Y, *et al*. Adaptive particle swarm optimization[J]. IEEE Transactions on Systems, Man, and Cybernetics, 2009, 39(6): 1362 - 1381.

[14] TANG K, LIU B, YANG J, *et al*. Double center particle swarm optimization algorithm[J]. Journal of Computer Research and Development, 2012, 49(5): 1086 - 1094. (汤可宗, 柳炳祥, 杨静宇, 等. 双中心粒子群优化算法[J]. 计算机研究与发展, 2012, 49(5): 1086 - 1094.)

(上接第 1332 页)

[10] SHI Y, KARATZOGLOU A, BALTRUNAS L, *et al*. CLiMF: Collaborative Less-is-More Filtering [C]// Proceedings of the 23rd International Conference on Artificial Intelligence. New York: ACM, 2013: 3077 - 3081.

[11] SREBRO N, JASON D M, TOMMI J. Maximum-margin matrix factorization [C]// Proceedings of the 22nd International Conference on Advances in Neural Information Processing Systems. Cambridge: MIT Press, 2005: 252 - 260.

[12] WEIMER M, KARATZOGLOU A, LE Q, *et al*. CofiRank - maximum margin matrix factorization for collaborative ranking [C]// Proceedings of the 22nd International Conference on Advances in Neural Information Processing Systems. Cambridge: MIT Press, 2007: 1593 - 1600.

[13] LIU N, ZHAO M, YANG Q. Probabilistic latent preference analysis for collaborative filtering [C]// Proceedings of the 2009 ACM International Conference on Information and Knowledge Management. New York: ACM, 2009: 759 - 766.

[14] LIU N, YANG Q. EigenRank: a ranking-oriented approach to collaborative filtering[C]// Proceedings of the 22nd International Conference on Research and Development in Information Retrieval. New York: ACM, 2008: 83 - 90.

[15] SHI Y, KARATZOGLOU A, BALTRUNAS L, *et al*. CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering[C]// Proceedings of the 5th ACM Conference on Recommender Systems. New York: ACM, 2012: 139 - 146.