LDA模型和列表排序混合的协同过滤推荐算法

王 涵 夏鸿斌
(江南大学数字媒体学院 江苏 无锡 214122)
(江南大学江苏省媒体设计与软件技术重点实验室 江苏 无锡 214122)

摘 要 基于排序学习的协同过滤推荐算法受数据稀疏性的影响,出现了推荐不准确性的问题。为此,文中提出了一种结合LDA主题模型和列表排序的混合排序学习协同过滤算法。该算法首先使用LDA主题模型对用户-项目评分矩阵建模,获取用户潜在低维主题向量来度量用户之间的相似度;然后通过列表排序学习函数为用户直接预测满足其偏好的排序列表。在Movielens和EachMovie两个真实数据集上的实验结果表明:该算法可以避免排序学习算法由于用户间共同评分信息过少引起的相似度计算不准确的问题,同时体现出了排序推荐的优越性,有效缓解了数据稀疏性带来的影响,提高了推荐准确度。

关键词 协同过滤,排序学习,列表排序,LDA主题模型
中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.09.032

Collaborative Filtering Recommendation Algorithm Mixing LDA Model and List-wise Model

WANG Han XIA Hong-bin
(School of Digital Media, Jiangnan University, Wuxi, Jiangsu 214122, China)
(Key Laboratory of Media Design and Software Technology of Jiangsu Province, Jiangnan University, Wuxi, Jiangsu 214122, China)

Abstract Rranking-oriented collaborative filtering is affected by the sparsity of data, which leads to the inaccuracy of recommendations. This paper proposed a hybrid ranking-oriented collaborative filtering algorithm based on LDA topic model and list-wise model. The algorithm uses the LDA topic model to model the user-item ratings matrix, and obtains the potential low-dimensional topic vector of the user, then measures the similarity between users with the topic vector. Next, the list-wise learning function is used to directly predict the total order of items that satisfies the users preference. The experimental results on the two real datasets of Movielens and EachMovie show that the algorithm can avoid the inaccuracy of similarity calculation between users caused by too little common score information, and at the same time reflect the superiority of learning to rank. It can effectively alleviate the effect of data sparsity and improve the accuracy of recommendation.

Keywords Collaborative filtering, Learning to rank, List-wise model, LDA topic model

1 引言

随着互联网信息技术的飞速发展,网络中迅速增长的信息数据引起的信息过载问题深刻影响着人们获取信息所面临的选择。协同过滤(Collaborative Filtering,CF)[1-2]作为一种应用最为成功的推荐技术,充分利用用户对项目的评分等历史行为数据进行推荐,能有效帮助消费者从海量信息中找到其感兴趣的信息。根据对已有算法的研究,协同过滤推荐算法可以依据其推荐方式的不同分为两大类[3]:传统的协同过滤推荐算法,即面向评分的推荐(rating-oriented CF);排序学习协同过滤推荐算法(ranking-oriented CF)。

传统协同过滤推荐算法可分为基于邻域的推荐(neighborhood-based CF)[4]和基于模型的推荐(model-based CF)[5]两大类。其中,基于邻域的推荐算法是通过历史评分信息获取用户或物品的相似度,得到邻居用户或物品集,再根据邻居用户或物品的历史行为预测目标用户对未知物品的评分,从而进行推荐。然而,根据预测到的评分产生推荐,并不能准确体现用户偏好。

基于邻域的排序学习协同过滤推荐算法很好地解决了传统基于邻域的协同过滤推荐算法预测不准确的问题。其以传统基于邻域的协同过滤算法为基本框架,结合机器学习方法对用户的历史评分信息进行训练,得到排序模型,进而直接预测得到满足目标用户偏好的最优排序列表,而不需要预测项目评分的中间过程[6-7]。排序学习可分为点级排序(point-wise)、对级排序(pair-wise)和列表级排序(list-wise)3类[8]。其中,列表级排序是直接对物品列表的排列顺序进行优化,通

到稿日期:2018-07-12 返修日期:2018-10-09 本文受国家科学支撑计划课题(2015BAH54F01)资助。
王 涵(1993-),女,硕士生,CCF会员,主要研究方向为推荐系统、机器学习,E-mail:W_hwang@163.com;夏鸿斌(1973-),男,博士,副教授,主要研究方向为计算机网络优化、社交媒体与数据挖掘、智能Web系统,E-mail:hbxia@163.com(通信作者)。

过构造损失函数或优化评价指标直接得到满足用户偏好的推荐列表，效果优于点级排序和对级排序。

虽然基于邻域的排序学习协同过滤推荐算法获得了更好的推荐效果，但是其排序模型的建立是基于用户间共同评分行为的物品集。随着推荐系统中用户和项目信息数据的不断增多，用户-项目评分矩阵越来越稀疏，实际中大部分用户的极少数评分，导致用户间共同评分的物品过少，影响了推荐的准确性。

为了提高数据稀疏性影响下基于邻域的排序学习协同过滤算法的准确度，同时突出列表级排序的优越性，本文提出了一种结合 LDA 主题模型和列表排序的混合排序学习协同过滤算法（LDA List-wise Collaborative Filtering，LDAList-CF）。LDA 模型是一种挖掘大型语料库中隐含主题的概率模型[9]，该模型通过低维隐含主题向量之间的相似性来度量两个高维文档之间的相似性[10]。本文首先借助 LDA 主题模型来构建用户-项目评分矩阵的伪文档，并使用 LDA 主题模型对评分矩阵进行间接模糊聚类，通过潜在主题向量连接用户和项目，在低维主题向量空间中计算不同用户间的相似度，从而得到目标用户的邻居用户集；然后，通过列表排序算法优化目标用户和邻居用户的排序概率间的交叉熵损失函数[3]，从而得到最终的排序列表。

2　相关工作

2.1　基于排序学习的协同过滤推荐算法

协同过滤算法是目前应用最广泛的推荐算法。为了解决传统协同过滤算法不能准确体现用户偏好的问题，排序学习协同过滤推荐算法得到了广泛的关注和应用。基于排序学习的协同过滤推荐算法也可以分为：基于邻域的排序算法和基于模型的排序算法[11-12]。本文着重研究基于邻域的协同过滤排序算法，下面将基于邻域的排序学习协同过滤算法分为 3 类进行介绍。

（1）点级排序：基于邻域的点级排序算法的本质仍是通过预测对未知项目的评分进行推荐。由 Breese 等提出的基于用户的协同过滤推荐算法[13]，通过余弦相似度计算用户间的相似度，然后根据邻居用户的历史行为进行评分推荐，其本质仍是传统协同过滤方法。该类算法被称为 Point-wise CF[3,11]。

（2）对级排序：侧重于考虑物品对之间的偏序关系，判断任意两个物品对之间的顺序关系，并将其转化为二元分类问题，通过分类函数进行学习，判断物品对之间的偏序关系，从而进行排序推荐。Liu 等提出的 EigenRank 算法[14]和 Wang 等提出的 VSRank 算法[15]都是基于用户的对级排序学习协同过滤算法。EigenRank 算法通过 Kendall Rank 相关系数计算共同评分物品集中任意物品对的相关性来度量用户之间的相似性，VSRank 算法采用向量空间模型来表示用户间共同评分物品对的偏序关系，两者都是通过采用贪婪聚合方法将预测的物品对之间的偏序关系进行聚合，从而得到最终的排序列表。该类算法被称为 Pair-wise CF。

（3）列表级排序：直接对物品列表的排序顺序进行优化，通过构造损失函数或者优化评价指标直接得到满足用户偏好的推荐列表，效果优于点级排序和对级排序。Huang 等提出的 ListCF 算法[3]使用排序概率模型来表示不同用户共同评分物品集中物品列表的排序概率分布，然后使用相对熵[16]来计算不同用户排序概率之间的相似性，再通过训练交叉熵损失函数得到最终的排序列表。该类算法被称为 List-wise CF。

2.2　LDA 主题模型

LDA 模型是一种主题概率生成模型，该模型认为在一篇文章的形成过程中，每个词的选择都是通过“以一定的概率选择某个主题，再从这个主题中以一定的概率选择某个词”的过程完成的[9,17]。因此，对于一篇文档，可以通过挖掘潜在主题发现文档与词汇之间潜在的相关性。通过将文档集中每篇文档的主题以概率分布的形式给出，实现了从高维词汇空间到低维主题向量空间的降维效果，从而根据低维主题向量实现文本分类等相关操作。这也使得 LDA 模型在文本主题分析、计算机视觉、基因序列识别和社会网络等领域得到了广泛应用。

近年来，很多研究者将 LDA 模型应用到推荐系统中进行研究。Zhou 等提出的 rating LDA model[18]是将评分信息添加到 LDA 模型中进行建模，认为对于用户的兴趣度描述，物品在用户中的高评分比例越高，该物品就越流行。Gao 等将 LDA 模型应用于传统协同过滤推荐算法，通过 LDA 模型挖掘用户和项目标签集上的潜在主题信息，结合评分矩阵共同计算用户和项目的相似度[19]。Peng 等充分利用评论文本的丰富信息，通过 LDA 模型挖掘物品潜在的属性面，并预测用户对物品属性面的评分，从而在属性面的层次上计算用户的相似度[20]。

本文研究发现，已有基于邻域的排序学习协同过滤算法有一个共同的特点：相似度的度量都是基于用户间共同评分行为的物品集。然而，随着用户-项目评分矩阵稀疏性的逐渐增大，用户间共同评分的物品逐渐减少，导致相似度计算不准确，从而严重影响了算法的推荐准确性。针对上述不足，本文将 LDA 模型应用于基于列表排序的协同过滤算法中，提出了混合排序学习协同过滤算法 LDAList-CF。该算法使用 LDA 模型对用户-项目评分矩阵建模，深层挖掘每个用户的潜在主题向量，进而使用低维潜在主题向量来计算用户之间的相似度，然后通过列表排序模型预测得到最终的排序列表。实验结果表明：该方法可以在体现列表排序优越性的同时，有效缓解数据稀疏性的影响。

3　混合排序学习协同过滤算法 LDAList-CF

本节详细介绍混合排序学习协同过滤算法 LDAList-CF，该算法是基于邻域的列表排序协同过滤推荐算法，分为相似度计算和排序预测两个步骤，具体框架如图 1 所示。

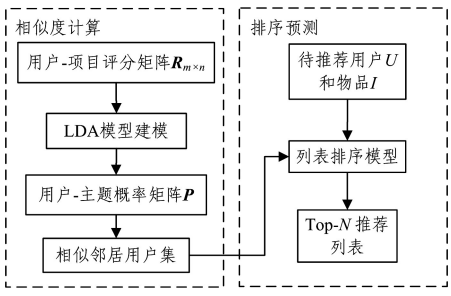


图 1　LDAList-CF 算法的框架

Fig. 1　Framework of LDAList-CF

3.1 结合 LDA 主题模型计算相似度

3.1.1 LDA 主题模型的构建

在 LDAList-CF 算法中，通过 LDA 主题概率模型优化计算用户的相似度。首先将 LDA 模型映射到用户-项目评分矩阵 $\boldsymbol{R}_{n\times m}$ 中，将评分矩阵转换为伪文档集合。用户集 U 对应语料库的文档集，一个用户 $u\in U$ 被视为一篇文档，项目 i 被视为词语，一篇文档中词语出现的次数转换到评分矩阵中可以表示为用户 u 对项目 i 的评分值 $r_{u,i}$，若没有评分行为则用 0 表示。假设用户 u 对项目 i_1 和项目 i_2 的评分分别为 1 和 3，则伪文档中用户 u 的内容表示为 $\{i_1,i_2,i_2,i_2\}$。将评分矩阵转换为伪文档的 LDA 图模型如图 2 所示。

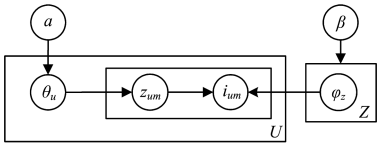


图 2 LDA 图模型

Fig. 2 LDA graph model

如图 2 所示，LDA 模型是一个三层贝叶斯模型。其中，α 和 β 分别是用户-主题概率分布 θ 和主题-项目概率分布 φ 的超参数，且满足 Dirichlet 分布。使用 LDA 模型对用户-项目伪文档进行建模，挖掘联系用户 u 和项目 i 的隐含主题，即用户的潜在兴趣度描述。根据图 2，每个用户 u 都可以表示为其隐含主题上的多项式分布 $\theta_u=(\theta_{u,1},\theta_{u,2},\cdots,\theta_{u,k})$，即用户的兴趣度分布；而每个主题 z 又可以表示为对应项目集上的多项式分布 $\varphi_z=(\varphi_{z,1},\varphi_{z,2},\cdots,\varphi_{z,m})$，即项目在主题上的属性分布。因此，可以将 LDA 模型训练过程描述为：用户 u 根据自己的主题(兴趣度)分布 θ_u，选择一个主题 z，然后根据主题 z 所对应的项目属性分布 φ_z 选择出一个项目。对用户集中的所有用户反复采样训练后，可以得到两个概率分布：用户-主题概率分布矩阵 $\boldsymbol{P}$ 和主题-项目概率分布矩阵 $\boldsymbol{Q}$。

通过 LDA 模型对评分矩阵构成的伪文档进行训练可以发现，用户-主题概率分布 $\boldsymbol{P}$ 表现为一个用户可能对多个主题有兴趣，主题-项目概率分布 $\boldsymbol{Q}$ 表现为一个物品的不同属性使其同时属于多个主题，将用户行为信息从高维评分矩阵空间转化到了低维主题向量空间，然后可通过低维主题向量计算不同用户的相似度。

3.1.2 相似度计算

本文中使用相对熵(KL 散度)[16]来计算用户相似度。相对熵一种是用来测量不同概率分布差异的方法。给定用户 u 和 v，则两者的相似度计算公式如下：

$$D_{KL}(\boldsymbol{P}_{\theta_u},\boldsymbol{P}_{\theta_v})=\sum_{k=1}^{t}\boldsymbol{P}_{\theta_{u,k}}\log_2(\boldsymbol{P}_{\theta_{u,k}}/\boldsymbol{P}_{\theta_{v,k}}) \tag{1}$$

其中，t 为主题个数，$\boldsymbol{P}_{\theta_u}$ 和 $\boldsymbol{P}_{\theta_v}$ 分别表示用户 u 和用户 v 的主题概率分布。又因为相对熵计算具有不对称性，因此可以通过相同的方式计算得到 $D_{KL}(\boldsymbol{P}_{\theta_v},\boldsymbol{P}_{\theta_u})$，然后重新定义用户 u 和用户 v 的相似度计算公式：

$$s(u,v)=1-(D_{KL}(\boldsymbol{P}_{\theta_u},\boldsymbol{P}_{\theta_v})+D_{KL}(\boldsymbol{P}_{\theta_v},\boldsymbol{P}_{\theta_u}))/2 \tag{2}$$

3.1.3 相似度计算的算法流程

相似度计算部分的算法实现如算法 1 所示。

算法 1 结合 LDA 主题模型计算相似度的算法

输入：用户-项目评分矩阵 $\mathbf{R}_{n\times m}$，伪文档超参数 α 和 β，最大迭代 max-Iteration_LDA

输出：用户 u 的邻居用户集 N_u

```
While(iter<maxIteration_LDA) do
  For u∈U do
  采样 θu~Dirichlet(α)
    For i∈I do
      For num∈rui do
        采样一个主题标签 zui~Multinomial(θu)
        采样一个物品 iu~Multinomial(φzui)
      End
    End
  End
更新用户-主题概率矩阵 P
For u∈U do
  For v∈U do
    s(u,v)=式(2)
  End
  筛选出每个用户的邻居用户集 Nu
End
```

算法 1 中，第 1－11 步是采用 Gibbs 采样训练 LDA 模型，通过反复采样得到用户-主题概率矩阵 $\boldsymbol{P}$；第 12－17 步通过式(2)计算不同用户之间的相似度，从而得到每个用户的邻居用户集。

以往提出的基于邻域的排序学习协同过滤推荐算法有一个共同的特点，即相似度的度量都是建立在不同用户间共同评分的物品集上。基于列表排序的协同过滤算法就是通过发现不同用户共同评分行为的物品集，建立排序概率模型来计算相似度。然而在实际应用中，评分数据的稀疏性导致了大部分用户间共同评分的物品非常少，造成了相似度计算不准确的问题。因为即使不同用户间共同评分的物品很少，也很难证明二者的兴趣度不相似，所以通过 LDA 模型将用户对项目的评分信息转化为低维空间中的潜在兴趣度描述。这样，即使共同评分的物品很少，但只要二者的潜在兴趣度相似，就能判断两用户相似，从而有效缓解数据稀疏问题对推荐准确度的影响。

3.2 列表排序预测

3.2.1 top-k 概率模型

top-k 概率模型是一种将用户表示为其评分物品列表排序的概率分布模型[3,21]，且只使用前 k 个位置的排序概率表示整个排序的概率。不同的排序概率分布代表了不同排序的可能性，概率越大，表示评分高的物品排在靠前位置的可能性越大。

给定一个包含 m 个物品的集合 I，将集合 I 上所有物品的一种有序排序表示为 $\pi=\langle\pi_1,\pi_2,\cdots,\pi_m\rangle$，其中 π_i 表示在该排序列表中第 i 位置上的物品，且当 $i\neq j$ 时，有 $\pi_i\neq\pi_j$。取与该排列前 k 个物品排序相同的所有排序，可构成 top-k 子集排序集 $G_k(i_1,i_2,\cdots,i_k)$，该集合包含了前 k 个排序相同的所有可能排序，Ω^I 表示集合 I 完整排序列表的所有可能排序，则有如下包含关系：

$$G_k(i_1,i_2,\cdots,i_k)=\{\pi\mid\pi\in\Omega^I,\pi_j=i_j,j=1,\cdots,k\} \tag{3}$$

如果 top-k 位置上的前 k 个物品排列不同，则二者的 top-

k 子集排序集不同，且用 G_k^I 表示集合 I 中所有可能 top-k 排序的集合。可以看到：集合 G_k^I 中的元素个数为 $m!/(m-k)!$，每个 top-k 排序子集中包含 $(m-k)!$ 种 Ω^I 中的排序。列表排序学习使用以下公式来计算一种 top-k 排序概率：

$$P(G_k(i_1,i_2,\cdots,i_k))=\prod_{j=1}^{k}\frac{\emptyset(r_{\pi_j})}{\sum_{l=j}^{m}\emptyset(r_{\pi_l})} \tag{4}$$

其中，r_{π_j} 表示对应位置上用户对物品的评分；$\emptyset(\cdot)$ 是严格单调递增函数，本文中使用 $\emptyset(r)=e^r$。top-k 排序概率 $P(G_k(i_1,i_2,\cdots,i_k))$ 等价于项目排列的前 k 个位置上的概率，并用来表示整个排序的概率，且有 $\sum_{G_k\in G_k^I}P(G_k)=1$。

3.2.2　预测排序列表

列表排序使用交叉熵作为损失函数来度量目标用户的邻居用户对应的评分列表排序概率和预测输出的排序概率之间的差异。给定目标用户 u，$I_u=\{i_1,i_2,\cdots,i_j\}$ 是待预测的物品集，N_u 是目标用户 u 的邻居用户集，且有邻居用户 $v\in N_u$，I_v 代表邻居用户 v 的评分物品集，则有公共物品集 $I_{u,v}=I_u\cap I_v$，该公共物品集中所有可能的 top-k 排序表示为集合 $G_k^{I_{u,v}}$，P_u 和 P_v 都表示在集合 $G_k^{I_{u,v}}$ 上的排序概率分布，则二者的交叉熵计算如下：

$$E(P_u,P_v)=-\sum_{g\in G_k^{I_{u,v}}}P_v(g)\log_2(P_u(g)) \tag{5}$$

其中，P_v 是相似邻居用户基于公共物品集的 top-k 排序概率分布，通过式(4)计算得出。而 P_u 表示目标用户 u 基于集合 $G_k^{I_{u,v}}$ 的待预测排序概率分布。为了保证所有概率分布之和为 1，对于任意一种 top-k 排序 $g\in G_k^{I_{u,v}}$，列表排序将概率 P_u 用以下公式表示并计算：

$$P_u(g)=\frac{\varphi_{u,g}}{\sum_{g'\in G_k^{I_{u,v}}}\varphi_{u,g'}} \tag{6}$$

其中，$\varphi_{u,g}$ 是待预测的排序概率值。

列表排序通过最小化目标用户和其所有邻居用户在公共物品集的 top-k 排序概率分布的交叉熵的加权值，为目标用户预测排序列表。基于交叉熵的损失函数计算如下：

$$\begin{aligned}&\arg\min_{\varphi_u}\sum_{v\in N_u}s(u,v)\cdot E(P_u,P_v)\\&\text{s.t. }\forall g\in G_k^{I_u}:\varphi_{u,g}\geqslant 0\end{aligned} \tag{7}$$

根据式(2)和式(5)，可以得到目标损失函数：

$$\begin{aligned}F(\varphi_u)&=\sum_{v\in N_u}s(u,v)\cdot E(P_u,P_v)\\&=-\sum_{v\in N_u}s(u,v)\cdot\sum_{g\in G_k^{I_{u,v}}}P_v(g)\log_2(P_u(g))\\&=-\sum_{v\in N_u}s(u,v)\cdot\sum_{g\in G_k^{I_{u,v}}}P_v(g)\cdot\Big[\log_2\Big(\sum_{g'\in G_k^{I_{u,v}}}\varphi_{u,g'}-\log_2(\varphi_{u,g})\Big)\Big]\end{aligned} \tag{8}$$

通过梯度下降法对式(8)进行优化，求导过程如下：

$$\frac{\partial F}{\partial\varphi_{u,g}}=\sum_{v\in N_u}\frac{s(u,v)}{\ln 2\cdot\sum_{g'\in G_k^{I_{u,v}}}\varphi_{u,g'}}-\frac{\sum_{v\in N_u}s(u,v)\cdot P_v(g)}{\ln 2\cdot\varphi_{u,g}} \tag{9}$$

不断迭代更新式(10)，得到最优概率值。

$$\varphi_{u,g}\leftarrow\varphi_{u,g}-\eta\frac{\partial F}{\partial\varphi_{u,g}} \tag{10}$$

其中，η 是学习速率。

3.2.3　排序预测的算法流程

排序预测部分的算法实现如算法 2 所示。

算法 2　预测排序列表的算法

输入：待推荐用户集 U，项目集 I_u，邻居用户集 N_u，最大迭代 maxIteration_prediction，误差阈值 e

输出：推荐列表 rank

1. For u∈U do
2.　While(t<maxIteration_prediction ‖ ε<e) do
3.　　初始化(φ_u^0)
4.　　For g∈$G_k^{I_{u,v}}$ do
5.　　$\varphi_{u,g}^t\leftarrow$更新($N_u$，s(u，v)，R)（式(10)）
6.　　$\varepsilon^+=\sqrt{\sum(\varphi_{u,g}^t-\varphi_{u,g}^{t-1})^2}$
7.　　End
8.　For i∈I_u do
9.　　P(i)←聚合($\varphi_{u,g}$)
10.　End
11.　rank(u)←排序 $P(i)_{i\in I_u}$
12. End

算法 2 中，第 1－7 步为损失函数梯度下降迭代训练过程，得到最优排序模型，从而求得排序概率值 $\varphi_{u,g}$，再通过式(6)求得目标用户 u 的待推荐物品的 top-k 排序概率分布。由于我们最终的目的是得到最优排序列表，因此第 8－10 步对求得的排序概率分布进行聚合：对于求得的所有 top-k 概率分布，将所有排序中排在 top-1 位置上的物品相同的概率值相加，从而求得每个物品排在 top-1 位置上的概率值。第 10－12 步通过将每个物品排在第一位置的 top-1 概率值进行排序，最终得到满足用户偏好的推荐列表。

4　实验结果及分析

4.1　实验数据集

我们使用 3 个被广泛应用于推荐系统研究的真实电影评分数据集：Movielens-100K[1)]，Movielens-1M 和 EachMovie[2)]。前两者是 Movielens 数据集的不同容量，体现了不同评分矩阵的稀疏度等级。Movielens 的评分范围是 1～5，EachMovie 的评分范围为 1～6。表 1 列出了 3 个数据集在用户、电影及评分等信息上的统计结果。

表 1　3 个数据集上的信息统计结果

Table 1　Information statistics results on three datasets

	Movielens-100K	Movielens-1M	EachMovie
用户(users)	943	6040	36656
电影(items)	1682	3952	1623
评分(ratings)	100000	1000209	2580222
评分/用户	106	165.6	70.4
评分/电影	59.5	253.1	1,589.8
稀疏度(sparsity)/%	93.7	95.8	95.7

表中数据集稀疏度(sparsity)等级的计算公式如下：

1) http://www.grouplens.org

2) http://grouplens.org/datasets/eachmovie

$$sparsity=1-\frac{ratings}{users\times items} \tag{11}$$

4.2 评测指标

在推荐算法中,比较常用的排序学习算法的效用评测指标是 NDCG(Normalized Discount Cumulative Gain)和平均准确率 MAP(Mean Average Precision)。这两个评测指标都是排序位置敏感的,能更好地体现排序算法的性能[6,8]。

4.2.1 NDCG

NDCG 是一种常用于排序学习推荐的多级评价指标,是对推荐列表排在前 k 个位置的物品是否满足用户偏好的评估。给定一个用户 u,其推荐列表中排在第 k 位置上的物品的 NDCG 值的计算公式如下:

$$NDCG_u@k=Z_u\sum_{i=1}^{k}\frac{2^{r_u^i}-1}{\lg(1+i)} \tag{12}$$

其中,$2^{r_u^i}$ 表示排在推荐列表 i 位置上的用户 u 对物品 i 的相关度等级;分母 $\lg(1+i)$是位置衰减函数;Z_u 是标准化因子,使得理想状态下的 NDCG 值为 1。待推荐用户集 U 的 NDCG 值等于所有用户的平均值,其值越大,推荐列表越能满足用户偏好,即算法的推荐效果越好。

4.2.2 MAP

MAP 常用于评估排序结果中相关文档的排序质量,体现了推荐列表和用户偏好的二元相关性。MAP 为所有用户推荐列表的平均准确率 AP(Average Precision)的均值。AP 的计算公式如下:

$$P@k(u)=\frac{R(u)\bigcap T(u)@k}{R(u)@k}\cdot l_k \tag{13}$$

$$AP(u)=\frac{\sum_{k=1}^{n_u}P@k(u)}{|R|} \tag{14}$$

其中,$P@k(u)$表示为用户 u 的排序列表中某一位置 k 的准确率(Precision);l_k 表示推荐列表位置 k 的物品与用户偏好是否相关,若相关为 1,否则为 0。$AP(u)$则表示一个用户 u 的推荐列表在所有位置上的准确率的均值,其中,$|R|$为推荐列表的长度。MAP 值越大,满足用户偏好的物品越靠前,推荐性能越好。

4.3 对比算法

将 LDAList-CF 算法与 3 种经典的基于邻域的排序学习协同过滤算法(PointCF[13],VSRank[15] 和 ListCF[3])进行实验比较。PointCF 是传统协同过滤算法,使用余弦计算相似度;VSRank 是对级排序学习算法,通过向量空间模型来表示用户对应物品对的偏序关系,从而计算相似度;ListCF 是列表级排序学习算法,通过将物品列表转换为排序概率来计算相似度。三者都是基于共同评分物品集的相似度计算。

4.4 实验结果及分析

本次实验是对结合 LDA 主题模型的混合列表排序协同过滤推荐算法的验证,采用 Gibbs 采样训练 LDA 模型,Gibbs 抽样是一种常用的统计推断手段。我们通过相关文献和大量的实验确定了 LDA 模型各参数的取值:主题个数 $t=20$,超参数 $\alpha=50/t$,$\beta=0.01$。设定各个算法的邻居个数为 200,以使得各算法可以达到最佳推荐状态。

4.4.1 top-k 排序概率的 k 值影响

本文提出的 LDAlist-CF 算法通过使用 top-k 概率模型产生排序概率来预测推荐列表。假设用户 u 对 m 个物品有评分行为,则通过 top-k 概率计算得到的概率分布集包含 $m!/(m-k)!$种不同的排序。特别地,当 $k=1$ 时,概率分布只有 m 种不同排序,可以发现 $k>1$ 时比 $k=1$ 时要多出 $(m-1)!/(m-k)!$倍排序,从而会消耗大量的计算时间。

因此,我们在 top-k 概率取不同 k 值时对算法的精确度和预测排序阶段耗时的影响做了验证。随机取 Movielens-1M 数据集的 1000 个用户,并且为每个用户推荐 20 个物品,当 k 取 1,2,3,4 时,算法的精确度和耗时如表 2 所列。

表 2 不同 k 值对 LDAlist-CF 算法精确度和时间的影响

Table 2 Effect of different k values on accuracy and time of LDAlist-CF algorithm

k	1	2	3	4
NDCG@5	0.741	0.7514	0.7604	0.7697
Time/s	4.2	43.1	777.8	18872.9

从表 2 中可以看出,当 k 值增大时,算法的精确度 NDCG@5 也在提高,例如,$k=2$ 时的精确度比 $k=1$ 时的精确度高 1.04%,$k=4$ 时的精确度比 $k=3$ 时的精确度高 0.93%,这证明了 k 值越大,算法的精确度越高。然而,精确度提高的同时,耗时也随着 k 值的增大在增加,尤其是 $k=4$ 时耗时比 $k=3$ 时耗时多出了约 24 倍,比 $k=1$ 时耗时多出约 4718 倍,因此精确度的提升以消耗大量的时间为代价,k 取更大值时,时间的消耗已经无法体现精确度的提升效果。因此,为了在节省时间的同时保障推荐效果,在之后的对比实验中都取 $k=1$。

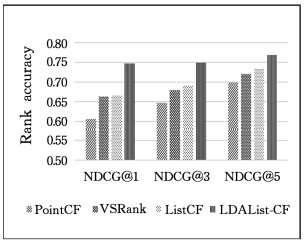
4.4.2 算法的精确度验证

我们使用 Movielens-1M 和 EachMovie 两个数据集对 4 组算法进行精确度验证,通过 NDCG 和 MAP 两种评测指标对实验结果进行精确度度量,并且随机取各数据集的 80%作为训练集,其余 20%作为测试集。验证结果如图 3 和图 4 所示。

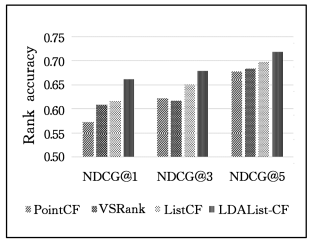
通过图 3 和图 4 可以发现:

(1)针对两种不同的评测指标,无论是在 Movielens-1M 数据集中还是在 EachMovie 数据集中,本文提出的 LDAList-CF 算法的精确度都要高于其他 3 种推荐算法。例如,在 Movielens-1M 数据集上,当取 NDCG@1,3 和 5 时,LDAList-CF 算法要比同是列表排序的 ListCF 算法高出 8.06%,5.71%和 3.44%。在 EachMovie 数据集上,当取 MAP@1,3 和 5 时,LDAList-CF 算法比 ListCF 分别高出 2.25%,1.44%和 0.72%。再结合表 1 可以看出,EachMovie 数据集中每个用户的平均评价行为更为稀疏,这也间接说明了 LDAList-CF 算法在一定程度上缓解了数据稀疏性的影响。

(2)LDAList-CF 算法和 ListCF 算法都是基于列表排序的协同过滤推荐算法,从两种评测指标度量得到的实验结果都可以看出:列表排序的精确度要明显高于传统基于点级排序的 PointCF 算法和对级排序算法 VSRank。例如,在 Movielens-1M 数据集上,列表级排序算法 LDAList-CF 和 ListCF 在取 NDCG@1 时分别高出 PointCF 算法 14.16%和 6.10%;在取 MAP@1 时,两种算法分别高出 VSRank 算法 4.26%和 2.34%,并且 LDAList-CF 算法比同是列表排序的 ListCF 算法的精确度更高。因此,LDAList-CF 算法既改善了数据稀疏性的问题,又体现了列表排序的优越性;在 EachMovie 数据集上也可以得出相同的结论。



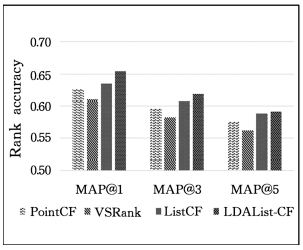
(a)Mvielens-1M



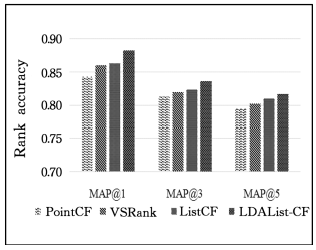
(b)EachMovie

图 3　使用 NDCG 在 Mvielens-1M 和 EachMovie 数据集上的精确度验证

Fig. 3　Accuracy on Mvielens-1M and EachMovie by NDCG



(a)Mvielens-1M



(b)EachMovie

图 4　使用 MAP 在 Mvielens-1M 和 EachMovie 数据集上的精确度验证

Fig. 4　Accuracy on Mvielens-1M and EachMovie by MAP

(3)与 NDCG 不同的是,MAP 体现为排序预测的位置越高,MAP@k 值越小,但两者都是对列表排序位置敏感的,其值越大,排列在靠前位置的物品越满足用户偏好。通过比较 4 种算法可以发现,列表排序算法表现出了位置上的优越性。例如,在 Movielens-1M 数据集上,当取 NDCG@1,3 和 5 时,LDAList-CF 高出 PointCF 14.16%,10.24%,7.02%,在取 MAP@1,3 和 5 时,LDAList-CF 高出 VSRank 4.26%,3.58%,2.19%,且 NDCG@1 比 NDCG@3 增幅更高,MAP@1 比 MAP@3 增幅更高,这说明越靠前的位置推荐效果越好。

4.4.3　算法的稀疏性验证

为了更好地证明本文所提算法对稀疏性的缓解效果,我们将 4 组对比算法应用在 3 个稀疏度等级不同的数据集上进行验证,并且使用 NDCG 和 MAP 两种评测指标进行实验结果的精确度度量。首先随机抽取 Mivielens-1M 数据集的 40%数据构造出稀疏度为 98.4%的数据集,抽取 80%数据构造出稀疏度为 96.7%的数据集,再使用稀疏度为 93.7%的 Movielens-100K 数据集。使用两种不同的评测指标在这 3 个稀疏度等级不同的数据集上进行实验验证的结果如表 3 和表 4 所列(括号内的值表示当前算法与 LDAList-CF 算法的精确度差值)。

观察表 3 可以发现:

(1)通过 3 个不同稀疏性的数据集得到的精确度可以看出,本文提出的 LDAList-CF 算法比其他 3 种算法的整体推荐效果都要好。例如,在稀疏度为 96.7%的数据集上,取 NDCG@1 时,LDAList-CF 算法比 PointCF,VSRank 和 ListCF 算法分别高出 10.53%,6.94%,4.25%。并且可以发现,无论数据的稀疏性如何变化,基于列表排序的 LDAList-CF 和 ListCF 算法都比基于对排序的 VSRank 算法和基于点排序的 PointCF 算法的精确度高。这充分说明,本文提出的 LDAList-CF 算法不仅可以体现列表排序的优越性,而且当数据变得越稀疏时,也起到了很好的缓解数据稀疏性的效果。

(2)将算法应用在 Movielens-100K 数据集时,LDAList-CF 在 NDCG@5 时比 ListCF 降低了 0.2%,而且在取 NDCG@1,3 时的提升幅度低于在其他 2 个数据集上的提升幅度。这是因为 Movielens-100K 数据集的数据量相对较少,仅是 943 个用户对 1682 部电影的评分,且稀疏度相对不大(93.7%),用户间共同评分的物品过少的性质不突出,LDAList-CF 算法的优越性也没有得到很好的体现。但当 NDCG@1,3 时,LDAList-CF 相较于 ListCF 分别提升了 2.52%和 1.07%,这说明即使在稀疏度相对不大的数据集上,LDAList-CF 也能取得相对更好的推荐效果。

表 3　4 种算法在 3 个不同稀疏度的数据集上以 NDCG 为评测指标的稀疏性验证

Table 3　Sparsity validation of four algorithms on three data sets with different sparsity by NDCG

数据集	NDCG@k	PointCF	VSRank	ListCF	LDAList-CF
Movielens-1M (98.4%)	NDCG@1	0.6138 (9.87%)	0.6527 (5.98%)	0.6791 (3.34%)	0.7125
	NDCG@3	0.6592 (5.78%)	0.6734 (4.36%)	0.7019 (1.51%)	0.7170
	NDCG@5	0.7012 (4.94%)	0.7179 (3.27%)	0.7444 (0.62%)	0.7506
Movielens-1M (96.7%)	NDCG@1	0.6194 (10.53%)	0.6553 (6.94%)	0.6822 (4.25%)	0.7247
	NDCG@3	0.6591 (6.62%)	0.6772 (4.81%)	0.7059 (1.94%)	0.7253
	NDCG@5	0.7086 (4.85%)	0.7195 (3.75%)	0.7459 (1.12%)	0.7571
Movielens-100K (93.7%)	NDCG@1	0.6451 (5.12%)	0.6740 (2.23%)	0.6711 (2.52%)	0.6963
	NDCG@3	0.6788 (3.01%)	0.6956 (1.33%)	0.6982 (1.07%)	0.7089
	NDCG@5	0.7258 (1.73%)	0.7343 (0.88%)	0.7451 (0.2%)	0.7431

表 4　4 种算法在 3 个不同稀疏度的数据集上以 MAP 为评测指标的稀疏性验证

Table 4　Sparsity validati on of four algorithms on three data sets with different sparsity by MAP

数据集	MAP@k	PointCF	VSRank	ListCF	LDAList-CF
Movielens-1M (98.4%)	MAP@1	0.7909 (3.84%)	0.7784 (5.09%)	0.8029 (2.64%)	0.8293
	MAP@3	0.7556 (5.48%)	0.7424 (6.80%)	0.7753 (3.51%)	0.8104
	MAP@5	0.7341 (6.28%)	0.7224 (7.45%)	0.7535 (4.34%)	0.7969
Movielens-1M (96.7%)	MAP@1	0.6748 (3.06%)	0.6900 (1.54%)	0.6991 (0.63%)	0.7054
	MAP@3	0.6558 (3.08%)	0.6561 (3.05%)	0.6723 (1.43%)	0.6866
	MAP@5	0.6341 (4.85%)	0.6350 (3.11%)	0.6527 (1.34%)	0.6661
Movielens-100K (93.7%)	MAP@1	0.7070 (4.14%)	0.6846 (6.38%)	0.7404 (0.80%)	0.7484
	MAP@3	0.6870 (5.26%)	0.6514 (8.82%)	0.7027 (3.69%)	0.7396
	MAP@5	0.6527 (5.65%)	0.6397 (6.95%)	0.6870 (2.22%)	0.7092

观察表 4 可以发现：

与将 NDCG 作为评测指标不同，在用 MAP 进行度量时，排序预测的位置越高，MAP@k 值越小。但是，以 MAP 为评测指标度量得到的实验结果与以 NDCG 作为评测指标进行度量的结果基本一致，即针对不同稀疏性的数据集，本文提出的 LDAList-CF 算法都要优于其他 3 种对比算法，且基于列表排序的 LDAList-CF 和 ListCF 算法具有更高的精确度。实验结果从另一种评测指标的角度体现了本文提出的 LDAList-CF 算法在体现列表排序优越性的同时，有效缓解了数据稀疏性问题。

结束语 本文提出了一种结合 LDA 主题模型和列表排序混合的排序学习协同过滤算法 LDAList-CF。首先，通过 LDA 模型对用户-项目评分矩阵构造的伪文档进行建模，发现联系用户和项目的潜在主题向量，以此来表达用户的兴趣度分布；然后，在低维兴趣度空间，根据不同用户的潜在主题概率分布，使用相对熵来度量两个用户之间的相似度，从而获取用户的邻居用户集；其次，在排序预测阶段，将物品列表转化为 top-k 排序概率，通过优化交叉熵损失函数来度量目标用户的邻居用户排序概率和预测概率之间的差异；最后，将预测概率进行聚合，得到满足用户偏好的完整排序列表。在 3 个不同数据集上的实验证明，LDA 模型聚类方法可以更好地挖掘出用户隐含兴趣，避免了不同用户共同评分物品过少引起的推荐不准确性问题，通过深层挖掘与目标用户更相似的邻居用户，再通过列表排序函数进行推荐，有效缓解了数据稀疏性问题，提高了推荐的准确度。

在今后的进一步实验中，我们考虑加入用户和项目的属性及标签信息，以增加推荐的多样性，并试图缓解推荐的冷启动问题。

参 考 文 献

[1] SCHAFER J B, DAN F, HERLOCKER J, et al. Collaborative Filtering Recommender Systems[J]. Acm Transactions on Information Systems, 2007, 22(1): 5-53.

[2] XING Y Y, XIA H B, WANG H. An Improved ALS Algorithm for Online Recommendation with Missing Data Modeling[J]. Computer Engineering, 2018, 44(8): 218-223.

[3] HUANG S, WANG S, LIU T Y, et al. Listwise collaborative filtering[C]// Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2015: 343-352.

[4] LINDEN G, SMITH B, YORK J. Amazon. com Recommendations: Item-to-Item Collaborative Filtering[J]. IEEE Internet Computing, 2003, 7(1): 76-80.

[5] HOFMANN T. Latent semantic models for collaborative filtering[J]. ACM Transactions on Information Systems, 2004, 22(1): 89-115.

[6] HUANG Z H, ZHANG J W, TIAN C Q, et al. Survey on Learning-to-Rank Based Recommendation Algorithms [J]. Journal of Software, 2016, 27(3): 691-713. (in Chinese)
黄震华，张佳雯，田春岐，等. 基于排序学习的推荐算法研究综述[J]. 软件学报，2016，27(3)：691-713.

[7] FANG C, ZHANG H, ZHANG M, et al. Recommendations Based on Listwise Learning-to-Rank by Incorporating Social Information[J]. Ksii Transactions on Internet & Information Systems, 2018, 12(1): 109-134.

[8] LU Y, CAO J. Research Status and Future Trends of Recommender Systems for Implicit Feedback [J]. Computer Science, 2016, 43(4): 7-15. (in Chinese)
陆艺，曹健. 面向隐式反馈的推荐系统研究现状与趋势[J]. 计算机科学，2016，43(4)：7-15.

[9] BLEI D M, NG A Y, JORDAN M I. Latent dirichlet allocation [J]. J Machine Learning Research Archive, 2003, 3(1): 993-1022.

[10] LIAN T, MA J, WANG S Q, et al. LDA-CF: A Mixture Model for Collaborative Filtering [J]. Journal of Chinese Information Processing, 2014, 28(2): 129-135. (in Chinese)
廉涛，马军，王帅强，等. LDA-CF：一种混合协同过滤方法[J]. 中文信息学报，2014，28(2)：129-135.

[11] SHI Y, LARSON M, HANJALIC A. List-wise learning to rank with matrix factorization for collaborative filtering[C]// ACM Conference on Recommender Systems, Recsys 2010. Barcelona, Spain, DBLP, 2010: 269-272.

[12] LIU J, WU C, XIONG Y, et al. List-wise probabilistic matrix factorization for recommendation [J]. Information Sciences, 2014, 278(9): 434-447.

[13] BREESE J S, HECKERMAN D, KADIE C. Empirical analysis of predictive algorithms for collaborative filtering[C]// Proceedings of the Fourteenth Annual Confe-rence on Uncertainty in Artificial Intelligence. Morgan Kaufmann Publishers Inc., 1998: 43-52.

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

[15] WANG S, SUN J, GAO B J, et al. VSRank: A Novel Framework for Ranking-Based Collaborative Filtering[J]. Acm Transactions on Intelligent Systems & Technology, 2014, 5(3): 1-24.

[16] KULLBACK S. Information Theory and Statistics[J]. Population, 1962, 17(17): 377-378.

[17] XIONG H X, DOU Y. Research on Tag Hybrid Recommendation Based on LDA Topic Model[J]. Library and Information Service, 2018, 62(3): 104-113.

[18] GAO N, YANG M. Topic Model Embedded in Collaborative Filtering Recommendation Algorithm[J]. Computer Science, 2016, 43(3): 57-61. (in Chinese)
高娜，杨明. 嵌入 LDA 主题模型的协同过滤推荐算法[J]. 计算机科学，2016，43(3)：57-61.

[19] ZHOU X, WU S. Rating LDA model for collaborative filtering [J]. Knowledge-Based Systems, 2016, 110: 135-143.

[20] PENG M, XI J J, DAI X Y, et al. Collaborative Filtering Recommendation Based on Sentiment Analysis and LDA Topic Model [J]. Journal of Chinese Information Processing, 2017, 31(2): 194-203. (in Chinese)
彭敏，席俊杰，代心媛，等. 基于情感分析和 LDA 主题模型的协同过滤推荐算法[J]. 中文信息学报，2017，31(2)：194-203.

[21] CAO Z, QIN T, LIU T Y, et al. Learning to rank: from pairwise approach to listwise approach[C]// International Conference on Machine Learning. ACM, 2007: 129-136.