考虑用户兴趣变化的概率隐语意协同推荐算法①

吴成超, 王卫平

(中国科学技术大学 管理学院, 合肥 230026)

摘 要: 推荐系统是人们从海量信息中获取对自己有用信息的一种有效途径, 在学术界和工业界都受到广泛关注. 协同过滤则是推荐系统领域最流行的算法, 目前很多协同过滤算法都是静态模型, 没有考虑到用户兴趣会随着时间而变化. 本文提出一种融合算法, 利用高斯概率隐语意(PLSA)模型提取出用户的长期兴趣分布, 然后结合用户评分时间窗捕获用户短期兴趣变化, 从而更准确的为用户做出推荐. 在Netflix和MovieLens数据集的上测试表明, 改进算法的预测评分准确率明显高于经典的基于用户相似度算法和 PLSA 算法.

关键词: 推荐系统; 协同过滤; 概率隐语意算法; 兴趣变化; 时间窗

PLSA Collaborative Filtering Algorithm Incorporated with User Interest Change

WU Cheng-Chao, WANG Wei-Ping

(Faculty of Management, University of Science and Technology of China, Hefei 230026, China)

Abstract: Recommend system is an effective method for people to get useful knowledge from mass information. It has attracted widespread attention in both academia and industry. Collaborative filtering (CF) is the most popular algorithm in the research of Recommend system. However most of current CF algorithms are static models, which do not take into account of user interest changing. The paper proposed a hybrid recommend method, which capture user's long-term interests with Gaussian probabilistic latent semantic (PLSA) algorithm, at the same time, capture user's short-time interests with rating window. The experimental results obtained on Movielens dataset and Netflix dataset clearly show that the new algorithm is more accurate than traditional user-based algorithm and PLSA algorithm.

Key words: recommender system; collaborative filtering; PLSA; interest changing; time window

1 引言

互联网飞速发展使得人类信息总量爆炸式增长, 出现了信息超载. 人们被海量的信息所淹没, 例如Amazon 上有数百万本书, Del.icio.us 上面有超过 10 亿的网页收藏[1], 人们要发现对自己感兴趣的内容是非常困难的. 个性化推荐系统的出现却改变了这一现状, 利用推荐系统记录用户的行为记录, 并结合用户、商品之间的关系, 用户甚至不需要做出任何显式搜索就会看到自己所感兴趣的商品. 因此推荐系统不管是在学术界还是工业界都受到了极大的关注和研究.

推荐系统方法主要包括基于内容算法和协同过滤两大类. 基于内容的推荐算法主要通过文本处理技术将用户兴趣用一个多维向量表示, 同时对项目也进行特征提取建立特征向量. 通过计算用户兴趣向量和项目特征向量之间的相似性来进行推荐[2].

协同过滤则包括基于记忆算法和基于模型算法以及各种融合算法. 基于记忆的协同过滤算法首先根据用户以往的行为记录, 计算用户之间、项目之间的相似性. 然后向用户推荐和其相似度大的用户所购买、评分的项目, 或者推荐和该用户以前购买项目相似度大的项目[3]. 然而实际的电子商务系统数据量非常庞大, 用户评分矩阵十分稀疏, 利用传统的基于记忆推荐算法会使准确率有所下降.

基于模型的协同推荐算法根据用户以前的评分和各种隐含的偏好计算出用户行为模型. 然后根据模型对用户的评分行为进行预测. 矩阵分解[4]推荐算法通

① 收稿时间:2013-09-28;收到修改稿时间:2013-10-24

过为 n 个用户和 m 个项目建立 k 维特征向量, 将一个 n×m 大小的评分矩阵转换成了 n×k 和 k×m 两个矩阵, 然后计算用户特征向量和项目特征向量的点积来对项目进行评分. 很多推荐算法都通过计算用户的全局相似度来寻找近邻进行推荐, 但是用户的兴趣可能只是在某一方面相似, 因此基于贝叶斯分类的推荐模型把评分分类, 分别采用不同的相似用户进行推荐[5]. 概率隐语意(PLSA)[6]方法也是一种基于模型算法, 该算法提取隐含变量来对用户偏好进行建模, 能获得比较高的准确率. 这些广泛使用的推荐算法大都是静态模型, 只是单纯的整合用户历史数据, 并未考虑用户兴趣变化情况[7]. 本文的改进算法通过引入用户评分时间窗对用户短期兴趣建模, 能区分出长期兴趣和短期兴趣并捕捉用户兴趣变化.

2 考虑用户兴趣变化的PLSA算法

2.1 高斯概率隐语意算法(PLSA)

概率隐语意是一种无监督学习模型, 在文本分析、图像识别方面有着广泛的应用, Hofmann 首先将概率隐语意模型引入推荐系统领域. 在 PLSA 推荐模型中引入隐含变量 $Z=\{z_1,z_2,...z_k\}$, 该模型认为一个用户选择了一个项目是由于某个隐含变量 z_k 导致的. 隐含变量既可以理解成具有某种相同兴趣偏好的社区, 也可以理解为某类相似项目所具有的共同属性. 假设推荐系统中有 n 个用户、m 个电影项目、k 个隐含类别, 则用户 u 对项目 y 评分为 v 的概率为:

$$P(v|u,y)=\sum_{z\in Z}p(z|u)p(v|z,y)$$

$p(v|z,y)$ 表示用户 u 属于某个社区 z 的概率, 表示社区 z 对项目 y 评分为 v 的概率. 高斯概率隐语意假设每个社区对项目 y 的评分服从高斯分布, 均值为 $\mu_{y,z}$, 标准差为 $\sigma_{y,z}$. 因此用户对项目评分概率可用一个混合高斯分布表示:

$$P(v|u,y)=\sum_{z\in Z}p(z|u)p(v;\mu_{y,z},\sigma_{y,z})$$

$$p(v;\mu,\sigma)=\frac{1}{\sqrt{2\pi\sigma}}\exp[-\frac{(v-\mu)^2}{2\sigma^2}]$$

参数 $p(z|u)$、$\mu_{y,z}$、$\sigma_{y,z}$ 分别用一个 n×k、两个 m×k 矩阵存储. 求解参数的方法是 EM 算法, Hofmann 的论文[8]中给出了详细的推导过程, 在此不再赘述, 仅简单描述如下:

(1) 定义损失函数:

$$R(\theta)=-\frac{1}{N}\sum_{<u,v,y>}\log P(v|u,y)$$

$$=-\frac{1}{N}\sum_{<u,v,y,z>}[\log p(v|z,y)+\log p(z|u)]$$

(2) E-Step(计算 z 的后验概率):

$$p(z|u,v,y)=\frac{p(v|z,y)p(z|u)}{\sum_{z'}p(v|z',y)p(z'|u)}$$

(3) M-Step:

$$p(z|u)=\frac{\sum_{<u',v,y>:u'=u}p(z|u,v,y)}{\sum_{z'}\sum_{<u',v,y>:u'=u}p(z'|u,v,y)}$$

$$\mu_{y,z}=\frac{\sum_{<u,v,y'>:y'=y}v\cdot p(z|u,v,y)}{\sum_{<u,v,y'>:y'=y}p(z|u,v,y)}$$

$$\sigma_{y,z}=\frac{\sum_{<u,v,y'>:y'=y}(v-\mu_{y,z})^2\cdot p(z|u,v,y)}{\sum_{<u,v,y'>:y'=y}p(z|u,v,y)}$$

E 步、M 步交替执行直到损失函数开始增加为止. PLSA 模型对于某个用户 u 和某个项目 y 的预测评分就是 u 对 y 评分概率 $p(v|u,y)$ 的期望:

$$\hat{r}_{u,y}=E[P(v|u,y)]=\int v\cdot P(v|u,y)dv$$

$$=\sum_z p(z|u)\int v\cdot p(v|y,z)dv=\sum_z p(z|u)\mu_{y,z}$$

2.2 用户兴趣变化分析与算法改进

PLSA 算法根据用户以前的评分记录提取了用户偏好分布, 即用户属于各社区 z_k 的概率, 这种提取更多的是从数量分布上考虑的, 并没有考虑时间因素. 现实生活中每个人的购买、评分行为是由长期兴趣和短期兴趣两部分决定的[9]. 长期兴趣反映用户的持久兴趣偏好, 相对稳定, 而短期兴趣随着日常生活环境的改变是会慢慢发生变化的, 以电影评分为例, 考虑下面这样一个例子:

表 1 某用户 100 个电影评分记录

	y1	y2	y3	y4	y5	…	y98	y99	y100
z1 科幻	√	√	√	√	√				
z2 文艺							√	√	√
z3……	√		√				√		

y1 到 y100 是某用户已经评分的 100 个电影, 按照

时间先后排列. y100 是最近才评分的电影, y1 是最早评分的电影. z_1, z_2, z_3 是隐含社区类别, 每个社区有着不同的兴趣偏好. “√”表示电影 y 属于社区 z, 或者说 y 包含某种被这个社区的人共同感兴趣的特征. 根据历史评分数据, 该用户属于 z_1 的概率最大, 因而按照 PLSA 算法所得 $p(z_1 \mid u)$ 会相对较高, 最终对其他项目评分预测也会更多的参考这个社区来预测. 但是从表中可以很明显看出用户最近对类别 z_2 有较高的兴趣偏好, 推荐系统应该向用户推荐和类别 z_2 相关性更大的电影.

现有解决兴趣变化的推荐算法多数是利用时间属性来对评分记录赋予不同的权重, 用户近期评分得到的权重较高, 因而在推荐时能起到更大的作用. 例如, 在基于物质扩散的推荐算法中, 通过对初始资源按照评分时间先后赋予不同的权重, 能显著提高推荐准确率[10]. 在 SVD 推荐算法中, 为评分项设置一个衰减函数, 使得算法在迭代过程中更多的考虑近期评分项目, 实验证明此算法能有效降低评分预测误差[11].

然而通过衰减函数来处理兴趣变化却不适用于 PLSA 算法, 因为 PLSA 算法的预测评分是根据用户所属各个社区的平均评分计算的, 而不是用户自身的评分. 因此, 本文提出一种新的处理用户兴趣变化的算法 DPLSA(Dynamic PLSA). DPLSA 算法为每个用户设置一个评分窗口 WINDOW, 在用户每次给一个新电影评分时, 向窗口中添加该电影并且去掉窗口中最早评分的电影, 如下图所示:

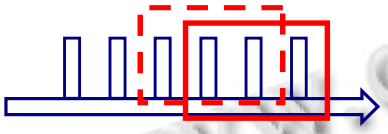

图 1 评分时间窗示意图

随着用户评分记录的增加, 窗口始终保存着用户近期评分电影的记录. 用户在某段连续时间内评分的电影有很大的相似性, 所以该窗口可以有效反映用户的短期兴趣. 结合窗口中的内容, 预测用户 u 对项目 y 的评分预测公式为:

$$\hat{r}_{u,y} = \alpha \cdot R_1(u,y) + (1-\alpha) \cdot R_2(u,y)$$

其中 $R_1(u,y)$ 是利用 PLSA 算法得出的预测评分, 反映用户的长期兴趣偏好. 而 $R_2(u,y)$ 是利用窗口中短期兴趣项目得出的评分预测. 通过两部分评分的融合不仅可以获取用户长期的兴趣偏好, 还可以及时发现用户兴趣变化. 参数 α 是平衡用户长期兴趣和短期兴趣权重的参数, 对于不同的数据集有不同的最优取值. 短期预测评分 $R_2(u,y)$ 公式如下:

$$R_2(u,y) = \bar{r}_u + \frac{\sum_{x \in WINDOW(u)} sim(y,x) \cdot (r_{u,x} - \bar{r}_x)}{\sum_{x \in WINDOW(u)} sim(y,x)}$$

$\bar{r}_u$ 为该用户的平均评分, $\bar{r}_x$ 表示其他用户对电影项目 x 的平均评分. $WINDOW(u)$ 表示用户 u 近期评分窗口中的所有电影项目. $sim(y,x)$ 表示待预测电影和评分窗口中电影 x 的相似度. 对于 $sim(y,x)$, 陈登科等人提出的融合 PLSA 模型中直接采用传统的评分余弦相似度进行计算[12]. 这种融合算法虽然能提高准确率, 但是运算时间太长, 实际上这种算法等于是 PLSA 模型和基于项目相似性两套模型同时运行. 与此不同, 本文采用各个社区 z_k 对电影 y、x 的平均评分组成的向量来计算相似性. 具体计算公式如下:

$$\mathrm{sim}(y,x) = \frac{U_y \cdot U_x}{\| U_y \| \cdot \| U_x \|}$$

U_y 是各个社区 z_k 对项目 y 的平均评分向量 $(\mu_{y,z_1}, \mu_{y,z_2} \ldots \mu_{y,z_k})$. U_x 是各个社区 z_k 对项目 x 的平均评分向量 $(\mu_{x,z_1}, \mu_{x,z_2} \ldots \mu_{x,z_k})$. 在实际的高斯 PLSA 模型中最佳隐变量个数一般只有几十个, 而两项目的共同评分用户很容易达到几百个, 可见利用社区对项目的评分向量来计算相似度, 运算量要小得多, 而且具有良好的扩展性, 不会随着用户和电影数量的增加而增加.

2.3 考虑用户兴趣变化的 PLSA 推荐算法描述

离线阶段:

(a) 根据用户历史评分记录, 利用 EM 算法求出 PLSA 模型参数 $p(z \mid u)$、$\mu_{y,z}$.

(b) 通过交叉验证, 求出最佳的窗口大小以及模型融合参数 α.

在线阶段:

(a) 利用 PLSA 算法求出用户对所有未评分项目的预测评分 $R_1(u,y)$.

(b) 从窗口获取近期评分项目, 求出用户对所有未评分项目的短期预测评分 $R_2(u,y)$.

(c) 融合两种预测评分, 得到用户 u 对所有未评分项目 y 的综合评分, 取其中得分最高的 N 个电影作为

用户的推荐结果.

(d) 当用户进行新的评分时, 记录用户评分信息, 并更新窗口内容.

3 实验结果与分析

3.1 数据预处理

实验数据来自于 Netflix 数据集和 MovieLens 数据集. Netflix 数据包含了 480000 名用户, 17000 部电影和近 1 亿条评分记录. 本实验从 Netflix 数据集中随机抽取了 1000 个用户、1825 个电影总计 89281 个评分项目, 数据稀疏度为 95.1%. MovieLens 数据集包含了 943 名用户 1682 部电影共 10 万条评分记录, 数据稀疏度为 93.6%. 把所有评分按照时间先后顺序排列, 然后抽取每个用户前 90%的数据作为训练数据, 后 10%的数据作为测试数据.

3.2 评价标准

预测精确度是评价推荐系统好坏的一项重要指标, 平均绝对误差(Mean Absolute Error, MAE)是一种常用的评价推荐系统精确度指标[13]. 本文亦采用 MAE 来度量评分准确性, 其计算公式为:

$$MAE = \frac{\sum_{<u,y>\in Test} | r_{u,y} - \hat{r}_{u,y} |}{N}$$

<u,y>为测试集中出现的用户、电影评分记录, N 为测试集记录总数, $r_{u,y}$ 为测试集中用户 u 对电影 y 的实际评分, $\hat{r}_{u,y}$ 为系统预测评分.

3.3 参数选择和实验结果

Hofmann 的论文[14]指出高斯概率隐语意算法隐变量个数在 20~50 之间均能获得较高的准确率. 针对本文的数据集, 通过测试表明隐变量个数设定为 20 能取得较高的准确率, 同时运行速度也较快, 因此后续实验中隐变量个数统一设定为 20. 为了考察参数对算法的影响, 首先固定评分窗口大小为 30, 在取不同的情况下, 利用 DPLSA 算法对 Netflix 数据集进行评分预测, 得到 MAE 结果如下:

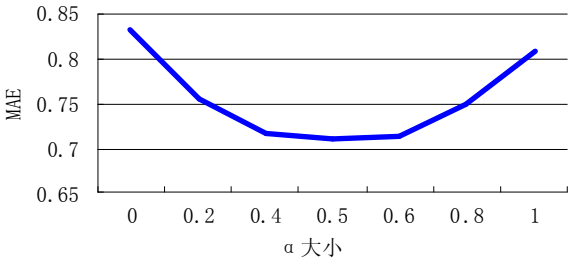

图 2 不同值对 MAE 的影响

对于 Netflix 数据集 α 取 0.4~0.6 之间可获得最高的准确率. 当 α 为 0 时表示预测完全由窗口中的数据决定, 当 α 为 1 时算法退化为原始的 PLSA 算法.

然后固定 α 为 0.5, 改变窗口大小得到的 MAE 结果如下:

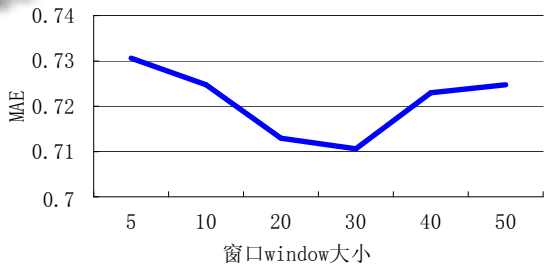

图 3 不同窗口大小对于 MAE 的影响

窗口过小时容易让噪音数据产生较大影响, 窗口过大会使短期兴趣和长期兴趣难以区分, 所以模型应该选择一个适当的窗口大小. 在 Netflix 数据集上窗口设为 20~30 左右时可获得较高的准确率. 反复试验, 尝试不同的 α 和窗口大小可以得到更加准确的推荐结果.

同样的方法在 MovieLens 数据集上进行试验并与经典的基于用户相似度协同推荐算法、原始的 PLSA 算法进行比较, 结果如下图所示:

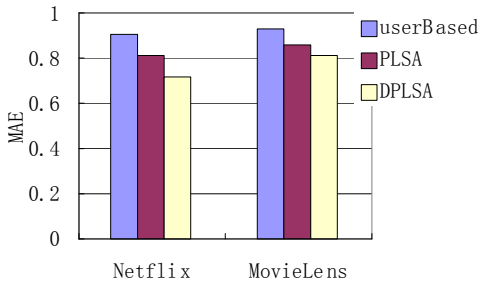

图 4 不同算法的精度比较

在 Netflix 数据集上, DPLSA 算法的 MAE 值比 userBased 算法下降了 20%, 比原始的 PLSA 算法下降

了 12%, 而在 MovieLens 数据集上分别下降了 13%和 5%. 在 Netflix 数据集上本文算法改进更明显, 这是由于两个数据集本身特点决定的, MovieLens 数据集时间跨度小, 而且用户评分大都集中在某些天[10], 评分记录分布不均匀, 更难捕捉兴趣变化.

4 结论

以往的很多协同推荐算法对用户兴趣进行建模时没有考虑到用户兴趣变化, 对所有评分数据一视同仁. 本文则通过设置用户近期访问时间窗, 在预测评分时, 将 PLSA 模型提取的用户长期兴趣偏好与时间窗内项目所反映的短期兴趣进行融合, 使得推荐系统能够跟踪用户兴趣的变化, 做出更加准确的推荐. 实际每个人兴趣是否变化, 变化快慢都不一样, 模型中却对所有的用户却采用了统一长度的时间窗, 在下一步的研究中将尝试采用动态时间窗来对用户兴趣进行更加准确的建模.

参考文献

1 刘建国,周涛,汪秉宏.个性化推荐系统的研究进展.自然科学进展,2009,19(1):1–15.

2 曹毅.基于内容和协同过滤的混合模式推荐技术研究[学位论文].长沙:中南大学,2007.

3 Sarwar B, Karypis G. Item-based collaborative filtering recommendation algorithms. WWW2007,285–295.

4 Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. IEEE Computer Society, 2009, 42(8): 30–37.

5 Miyahara K, Pazzani MJ. Collaborative filtering with the simple bayesian classifier. 6th Pacific Rim International Conference on Artificial Intelligence. 2000, 1886. 679–689.

6 Hofmann T. Latent class models for collaborative filtering. Proc. of the International Joint Conference in Artificial Intelligence. 1999. 688–693.

7 王卫平,刘颖.基于客户行为序列的推荐算法.计算机系统应用, 2006,15(9):35–38.

8 Hofmann T. Latent semantic models for collaborative filtering. ACM Transactions (TOIS), 2004, 22(1): 89–115.

9 Liang X, Yuan Q, Zhao S, Chen L, Zhang X, Yang Q, Sun J. Temporal recommendation on graphs via long- and short-term preference fusion. SIGKDD. 2010.723–732.

10 丁建勇.基于时间特性的网络结构推荐[学位论文].成都:电子科技大学,2009.

11 顾申华.结合奇异值分解和时间权重的协同过滤算法.计算机应用与软件,2010,27(6):256–259.

12 陈登科,孔繁胜.基于高斯 pLSA 模型与项目的协同过滤混合推荐.计算机工程与应用,2010,46(23):209–234.

13 Herlocker JL, Konstan JA, et al. Evaluating collaborative filtering recommender systems. ACM Trans. (TOIS), 2004, 20(1): 5–53.

14 Hofmann T. Collaborative filtering via gaussian probabilistic latent semantic analysis. Proc. of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval. 2003. 259–266.