基于BPR-MF算法的推荐系统设计文档（V0.1）

1. 背景分析

基于邻域的协同过滤算法和矩阵分解模型是目前主流的两种推荐算法。基于邻域的协同过滤算法主要分为User-based CF和Item-based CF两种，其中User-based CF适用于用户数较少，item数量较多并且更新频繁的系统（如新闻、博客、微博等）；而Item-based CF适用于item数量相对稳定，且用户数量远大于item的系统（如绝大多数电商）。我们目前在电商的动态创意推荐逻辑中采用的就是item-based CF算法。

Item-based CF的局限性在于对较为稀疏的user-item矩阵表现较差，而且模型本身没有考虑到item的热门程度信息。矩阵分解模型可以较好地解决矩阵稀疏性问题，并通过加入item-bias或者采用leaning-to-rank的方式training使得模型倾向于推荐同一类目中较为热门的商品。因此，我们通过BPR-MF（Bayesian Personalized Ranking for Matrix Factorization）对现有的推荐算法进行改进。

1. 算法设计

BPR算法将用户对物品的评分处理为一个pair对的集合<i,j>，其中i为评分为1的物品，j为评分为0的物品。假设某用户有M个“1”的评分，N个“0”的评分，则该用户共有M\*N个pair对。这样数据集就由三元组 <u,i,j>表示，该三元组的物理含义为：相对于物品“j”，用户“u”更喜欢物品“i”。数据假设：(1) 每个用户之间的偏好行为相互独立; (2)同一用户对不同物品的偏序相互独立.

问题的优化目标函数可以写为：

其中=,为sigmoid函数 ,

求导得到：

其中

BPR-MF算法的伪代码：

Initialize:

For any user *u* and feature *f*,

For any item *i* and feature *f*,

Repeat:

Draw from training data set

Until Convergence

1. 实验结果
2. 离线实验
3. 试验数据

采用7天（2015-05-20到2015-05-26）的用户浏览日志，利用top 1,000,000用户和top 10,000,000商品构建user-item矩阵，在其中将10%的浏览记录删除作为测试数据，剩余的90%的作为训练数据。

1. Candidate生成

利用w-cosine工具计算item-item相似度，item-based collaborative filtering(协同过滤)计算user-item相似度。每个user取top 10个item作为备选集合。

1. 实验设置

对每个user, 将备选集合中的item进行排序，选取top 6计算recall值。Baseline采用CF算法原始的score排序；BPR（Bayesian Personal Ranking）算法将user和item分别表示成k维的向量, 训练出向量的取值后，user和item的相关度可以通过对应向量的点乘计算。参数设置：训练步长0.005，正负样本采样比1:30，初始值服从N(0, 0.01), 维度5，combine参数0.01, 正则项0.000001，迭代轮数10。

评估指标为Top-6 recall, 结果如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | CF | BPR | Combine | RealTime |
| 1095(唯品会) | 0.0688 | 0.0696 | **0.0706** | **0.0706** |
| 1052(我买) | 0.1117 | 0.1206 | **0.1235** | 0.1232 |
| 1631(丽芙) | 0.1744 | 0.1610 | 0.1783 | **0.1794** |
| 955(当当) | 0.0962 | 0.1043 | **0.1053** | 0.1040 |
| 1352(寺库) | 0.0762 | **0.0870** | 0.0847 | 0.0850 |

其中RealTime Setting不显式表示user的vector, 而是用user历史浏览过的item vector作为user vector的估计值，从而方便进行实时更新。

1. 第一阶段在线实验

离线对模型定时更新，每天开始投放之前更新一次。测试时，只针对当日未有过浏览行为的用户进行A/B test。测试结果如下表所示。其中粗体字为BPR-MF+CF算法评估结果（算法版本1），其余为CF-baseline算法评估结果（算法版本0）；可以看到在唯品会和我买两个广告主上，多日CTR平均有较为稳定的提升，有些日期提高达20%左右。



1. 第二阶段在线实验

对所有曾经有过浏览单品行为的user进行全量A/B test。将user按照userId取模随机分成3份, 其中“19”和“11”两份采用BPR-MF+CF算法作为推荐逻辑，“10”采用CF-baseline算法作为推荐逻辑。实验结果见下表。目前的模型参数是通过唯品会离线测试数据优化确定的，且对所有广告主采用一致的参数进行测试。可以看到目前模型在唯品会、寺库和蜜芽宝贝这几个广告主上有稳定的提升，而在我买和有货两个广告主上表现不稳定。下一步将单独为特定广告主优化参数，检验是否有进一步提升空间。



1. 系统设计

参考Netflix实时推荐系统的设计，将系统分为Offline, Nearline, Online三部分。

1. Offline

Offline部分主要负责算法模型的离线训练和更新。目前的实现是每隔2-3个小时更新一次，步骤如下：(1)通过Hadoop收集最近7日的浏览商品记录构建浏览矩阵并加载到内存; (2)单机版的BPR-MF算法训练得到user vector和item vector的结果; (3)计算Cosine Similarity得到每个user推荐的Item及对应score, 然后与CF算法得到的结果和历史浏览过的商品进行combine, 构建出最终推荐列表并更新到Redis。

当广告主数目较多的时候，可能导致多个广告主的更新程序同时将浏览矩阵加载到内存，使得单机内存负载量过大甚至溢出。目前对训练进程加入了进程锁保护，同时只允许最多三个广告主矩阵被加入内存，在目前的广告主数目规模下可以稳定地运行。随着广告主数目的不断增加，可能单机的模式会造成较大延迟，以后将考虑把训练模块改为Spark实现。

1. Nearline

仅通过离线模式对BPR模型进行训练更新会造成较大的数据延迟，影响推荐算法的性能。实践中，当一个User浏览了一个新商品之后，我们希望立刻对user的vector进行重新计算，使得下次的推荐就能够考虑到他的新行为。因此，user vector的更新应该是实时或者准实时的。假设用户之前浏览了N个商品，user vector为X, 新浏览的item vector为Y, 那么更新方程可以写为：

Nearline的准实时更新逻辑可以通过流式处理框架实现，如Storm或者Spark Streaming，将用户新增的浏览记录实时地通过流的形式发送到更新模块，模块从Redis里面取到对应的vector并将计算结果更新至Redis.

1. Online

Offline和Nearline计算的推荐列表会保存一个时间戳。当一个推荐请求发生后，在Online部分先从Redis当中取到最新的缓存结果以及对应的时间戳，如果在该时间戳之后用户发生了新的浏览单品行为，会基于该商品对推荐列表进行增量打分（该商品以及与该单品相似的商品相应的推荐得分获得提升），最后将推荐结果反馈给用户。

1. 算法优化
2. 离线评测指标和Online目标不一致

目前离线的评估指标采用预测浏览商品集合的Top 6 Recall, 而Online的优化目标是提升CTR。评估指标的不一致加大了参数优化的难度，离线最优的指标在线测试时未必是最优的，而通过A/B test重新对参数进行调优也比较困难。

可以尝试将离线评测指标改为点击商品集合的Top 6 Recall, 再选取最优的参数进行Online A/B test，观察效果是否能够进一步提升。同时在训练时可以加大点击商品的权重，使得在BPR算法中“点击商品”>“浏览商品” > “未浏览商品”，观察能否能提高点击商品集合的Top 6 Recall。

1. 推荐多样性

如右图所示，很多相同的商品由于size、颜色等不同具有不同的item ID, 导致推荐的时候商品多样性很低，影响用户体验。

1. 非单品访客的推荐

目前在访客找回逻辑中，有过浏览单品行为的访客流量大致能覆盖总流量的50%-80%， 还有相当一部分的访客没有过浏览单品的行为，无法基于目前的推荐逻辑进行商品推荐。 非单品访客的推荐算法考虑如下两种算法作为baseline:

1. 热门程度 (Popularity)

为所有用户找到同一批热门商品，每次随机选取一个子集进行推荐。

1. Item-based CF

将用户浏览过的URL当作item统一处理，通过w-cosine可以计算出每一对<URL, Item> pair的相似度；对于非单品访客，通过已浏览过的URL推荐与该URL相似的商品。

1. BPR-MF + CF

用户浏览过的URL vector的平均值作为User vector, 然后再与Item Vector点乘的结果作为推荐打分。利用BPR-MF算法基于用户浏览单品的行为进行训练，得到URL vector和Item vector。对于非单品访客，同样可通过历史浏览过的URL vector的平均值计算user vector, 然后与各个备选的item vector计算相似度。这里也需要一个备选item集合，以避免在全量item列表上面计算相似度。我们采用Item-based CF的结果作为初始集合, 在此基础之上利用BPR模型的打分结果进行re-rank。

1. 下一步工作计划
2. Online参数调优，算法优化（P0）
3. Nearline准实时更新逻辑（P1）
4. 非单品访客的推荐（P2）
5. 推荐多样性（P2）
6. 将BPR-MF训练模块改为Spark实现（P2）
7. Milestones
   1. 7.14~ 7.20：Online参数调优；
   2. 7.20~7.30: 算法优化，上线；
   3. 7.30~8.10：Nearline准实时更新逻辑实现；