1绪论

1.1研究背景及意义

1.1.1 O2O商业模式的巨大市场

O2O商业模式的出现，是互联网服务于传统行业的一个例子。【1】Online To Offline，意为线上到线下，以实体店为依托，以互联网为平台。商家通过线上的推广活动，吸引用户在线下进行消费行为。通过互联网服务，发布新品消息，折扣消息等，吸引用户。而对于用户而言，可以通过线上的信息反馈，了解商家的服务质量，资源环境等一系列信息。

与之相对应的，是一系列服务网站的产生。团购网站被认为是最早的O2O模式，其中以美团为代表。

据中国电子商务报告数据显示，2017年全国电子商务交易额为29.16万亿元，同比增长11.7%，其中，中国本地生活服务O2O整体市场规模达9992亿元，相比去年增长71.5%。在交易额上，口碑和美团点评分列行业第一第二。

1.1.2 推荐系统在各行各业中的应用

推荐系统在电子商务网站中，更多是扮演销售人员的角色，为用户提供建议，帮助用户完成决策行为。而个性化推荐，可以根据用户的兴趣特点，还有历史的消费行为，为用户进行“私人定制”。完成对用户的推荐行为。

O2O商业模式的Online模块，可以采集用户数据，包括用户的位置信息，购买行为。分析用户数据，找到用户在Offline下，POI之间转移的特点。对用户进行推荐行为。提高用户体验。

推荐是信息过滤的一种形式，它利用用户过去的行为和用户相似点生成一个信息项的推荐列表，该列表为用户进行个性化定制。

1.1.3 推荐系统，在POI推荐中的应用

1.2关键技术国内外相关研究

推荐算法概括来说，可以分为基于内容的推荐，协同过滤推荐，混合推荐，还有就是基于规则的推荐。其中，基于内容的推荐，一般依赖于NLP，通过挖掘文本的特征向量来得到用户的偏好，这类推荐算法，一般都有较好的解释性。混合推荐，类似机器学习中的集成学习，一般都是通过多个不同推荐算法的结合，得到推荐结果，效果理论上比单一的算法要好很多，但是算法的复杂度较高。而对于协同过滤算法，一般又可以分为三种类型，分别是基于用户的协同过滤，是基于物品的协同过滤，还有就是基于模型的协同过滤。基于用户的协同过滤算法，他的思想是，具有相同兴趣爱好的人，喜欢的东西也可能会相同，一般用矩阵来表示用户行为，矩阵项为用户对于物品的打分，最后，计算两个矩阵的相似度，来得出两个用户之间的相似度的。但是，这种推荐算法存在明显的缺点，随着用户量的的增多，用户的相似性的计算会越来越困难，算法的时间复杂度还有空间的复杂度都会很高。基于物品的协同过滤算法，他的思想是，向用户推荐他之前选择的相似的物品，这个地方是根据所有用户对于物品的打分，来计算物品之间的相似度，他的优点在于基于用户的历史行为，对于推荐，有合理的解释，同样的，在计算物品相似度的过程中，随着物品种类的增多，物品之间相似度的计算代价会变大。而且无法发掘用户的隐形爱好。

对于基于模型的协同过滤，我们可以用关联算法，聚类算法，分类算法，回归算法，矩阵分解，神经网络，还有图模型等来实现协同过滤。

矩阵分解，在推荐系统中，有很多的用户和物品，其中，已有的是部分用户对部分物品的评分，我们要做的就是，根据已有的评分数据，去预测目标用户对于未评分物品的评分，然后根据评分生成推荐列表。传统的SVD（奇异值分解）要求矩阵是稠密的，所以一般会对评分矩阵进行补全操作，一般是全局平均值或者是物品的平均值，然后再进行分解降维。但是在实际的生产中，用户数和物品数都是很大的，对一个千万级矩阵进行SVD分解，时间会非常的长。而FunkSVD采用线性回归的思想，用均方差作为损失函数，将评分矩阵分解成两个矩阵P和Q

（https://www.cnblogs.com/pinard/p/6351319.html）

FunkSVD的改版，BiasSVD在评分系统中添加了三部分的偏置因素，在某些场景会比FunkSVD表现好。SVD++算法，在BiasSVD算法上进一步加强，考虑用户的隐私反馈。目前张量分解和分解机分解是矩阵分解推荐方法的趋势

贝叶斯个性化排序（BPR）也用到了矩阵分解，但是和FunkSVD有很大的不同，在实际的推荐场景中，我们要在千万级的商品中推荐个位数的商品给用户，这个时候，需要的是为每个用户生成一个商品的喜好排序列表，让少数优先级高的商品排序靠前。BRP是基于矩阵分解的一种排序算法，和FunkSVD之类的算法相比，它不是生成全局的分解矩阵，而是针对每个用户，生成商品的喜好排序列表。

SimRank基于图论，如果两个用户相似，则这两个用户相关联的物品也类似，如果两个物品类似，则这两个物品相关联的用户也类似。SimRank++算法，对SimRank算法做了两点改进，第一添加了边的权值，第二考虑了子集节点相似度。对于SimRank算法，如果用户和物品量非常大，涉及矩阵运算的计算量非常大，加上迭代的方法，会花上很长的时间，一般会用两种方法来加快求解速度，一种是利用Hadoop的MapReduce或者Spark来将矩阵运算并行化，加速求解过程。第二种是利用蒙特卡洛法（Monte Carlo，MC）模拟，这里用两个随机游走者分别从两个节点出发，用最后相遇的总时间的期望函数来表示两个结点的相似度。算法的时间复杂度会大大降低，但是有随机性，结果的精度不高。

分解机（Factorization Machines，FM）

区域影响因子的确定，可以很大程度上降低计算成本，也可以丢弃噪声信息

1.2.1 连续的POI推荐

【2】中cheng等人第一次阐述了连续POI推荐问题，然后提出了因式分解个性化马尔可夫链（FPMC）来解决连续推荐的问题。FPMC最开始是用来解决基于当前的下一个的推荐问题的。Cheng等人在推荐兴趣点的过程中，在FPMC算法中也加入了地理影响因子。因为在众多的POIs中，用户通常只会在一段时间内在很小的一部分POIs上完成签到行为。所以，签到数据往往有很大的稀疏性。针对这种情况，He等人【3】不仅用马尔可夫链来推断用户在POIs上的偏好，而且根据用户的签到行为，按照POIs的类型或者签到时间，来将用户进行分组。来解决数据稀疏的问题。Zhao等人认为【4】顺序的签到行为受时间的影响。因此，他们使用时间因素来衡量顺序签到行为的重要性，并相应的提出了一种连续的POI推荐方法。

Feng等人【5】利用距离嵌入（metric embedding），将POI映射到低维度的欧氏空间中，用两个POI的欧式距离来确定二者的序列关系，最后综合序列信息和个人喜好，提出个性化距离嵌入排名算法（PRME）。此外，Liu等人【6】根据POI的访问顺序探究POI的上下文信息。首先，他们利用原本为自然语言处理（NLP）设计的Skip-Gram模型【7】，探究POI访问顺序的影响。然后，基于【8】中提出的个性化推荐模型，将访问频率带入模型。

1.2.2

1.3研究内容

1.4论文的组织结构

2问题分析

2.1 问题定义



连续推荐中，问题可以归纳为下面几点

U和L分别代表用户和POIs的集合，Lu表示用户u在POIs中已经访问过的POI集合 ，对于给定的q(u,lc,tc),其中u∈U,为目标用户，lc∈Lu是用户签到地点，tc是当前时间，连续POI推荐的问题在于推荐N个用户u没有去过的POIs，可以用式子表示为Ru，N。其中对于每一个POI， l∈Ru，N，满足下面条件。

1. l和lc之间的距离要小于等于距离阈值d。
2. 用户u会在时间段[tc,tc+τ]内，在l处完成签到行为。

2.2 数据库描述

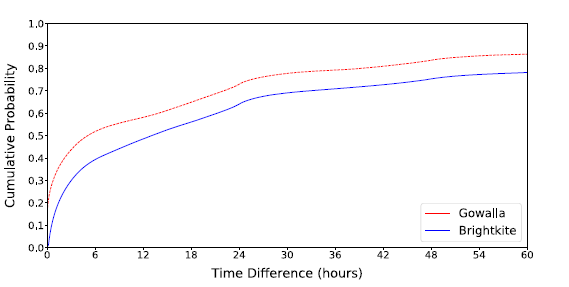
本次实验的数据集为Gowalla和Brightkite。因为一些签到记录的信息不足，不活跃用户的签到记录和那些不受欢迎的POIs记录，会被从数据库中移除，因为这些记录会严重影响推荐的性能，在本文中，数据满足以下几点，会被删除。

1. 少于80个签到记录的用户的签到行为
2. POIs中，少于五个用户签到的位置
3. 用户中少于五个好友的

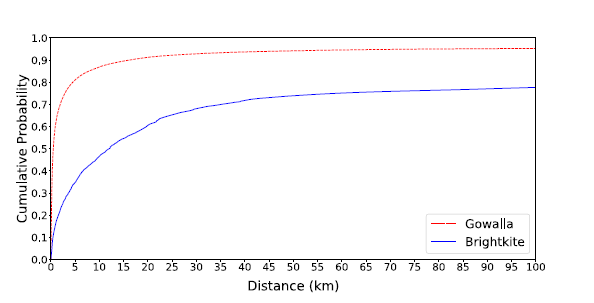
通过数据清理，数据集中的数据表示如下

|  |  |  |
| --- | --- | --- |
| Dataset | Brightkite | Gowalla |
| #Users | 3178 | 6415 |
| #Locations | 6792 | 44938 |
| #Check-ins | 291655 | 819283 |
| #Edges | 45308 | 73410 |
| Periods | 2008.04-2010.10 | 2009.02-2010.10 |

在连续的POI研究中，连续POI的签到在很大可能上会受到上一时间段签到行为的影响，如下图显示，Gowalla和Brightkite数据集中两个连续签到记录的时间差在所有签到记录中的比例，在Gowalla数据集中，大约52%的连续签到行为发生在6个小时内，超过72%的数据发生在24小时内，在Brightkite数据集中，超过40%的连续签到行为发生在6小时内，超过60%的发生在一天内。这里可以看出，Brightkite数据集在时间域上比Gowalla数据集更加的稀疏



除了时间差异之外，距离差异也会影响用户签到行为，下图是连续签到行为发生在6小时内，签到地点距离差的分布图，从图中可以看到，在Gowalla数据集中，超过90%的连续签到行为发生在距离当前位置距离15km的位置内，而在Brightkite中，也有超过50%的数据。很显然，Brightkite数据集在空间领域也比Gowalla数据集更稀疏。



2.2 模型架构



2.2.1用户偏好

这个地方使用基于用户的协同过滤（User-based CF）来计算目标用户u对于POIs的偏好得分。这里有两个用户v和u，让cu,l=1表示用户u已经在POI l处进行过签到行为，否则cu,l=0，根据用户v和u的历史签到记录，用下面的等式表示用户u和用户v的相似度。

 （1）

U'是用户u在签到行为上相似的用户的集合，用户u对POI l偏好程度的分数可以用下面的等式表示

 （2）

2.2.2 区域影响

与传统的POI推荐不一样，连续POI推荐很大程度上会考虑推荐POI和当前用户所在位置的距离，如之前数据集中的数据显示，用户选择下一个POI时，很大程度上会选择距离自己较近的，特别是具体到商圈中，用户在某个商圈中的消费行为，很大程度上都在一个区域里面。这也就意味着，用户很小的可能会选择一个比较有趣，但是却距离自己现在位置很远的地方，因此，我们可以将距离远近作为一个影响因子，放在连续的POI推荐中，如图所示，将地理位置划分为很多大小相等的格子，以用户u当前位置lc为圆心，阈值d为半径作圆。圆与网格相交的阴影部分，我们称之为用户u的临近网格单元。这来源于生活实际，一个区域内，有很多人进行签到行为，那么这个区域必定是一个受欢迎的区域，与之相对应的，这个区域里面的POIs，也会更受欢迎。就好像每个每个商圈中的店铺一样。



这里我们让Checkins(gi)作为网格单元gi中所有签到记录的总和，那么网格gi的受欢迎程度可以用等式表示为

 （3）

其中Gs是用户u的临近网格单元的集合

当用户u在某个网格单元gi中，有很多的签到行为。那么有很大的可能gi区域是用户u最喜欢的区域，用户u有很大的可能会在这个gi中的其他POIs，进行签到行为。就好像用户经常活动，以及发生消费行为的，都会是某个固定的商圈。这个地方，我们让Checkins(gi,u)表示用户u在网格gi中所有POIs的签到记录总和。我们可以用等式来表示这种影响。

 （4）

正如数据集所显示的，用户往往倾向于在距离当前位置较近的POIs发生签到行为，因此，我们可以推断出，用户倾向于在当前位置所有的网格内的POIs进行签到行为，我们可以将这种影响表示为下面的等式

 （5）

我们将三种影响因子混合在一起，通过线性方程组将网格gi得分表示为如下等式

 （6）

其中α,β还有γ是满足下列条件的常量

 （7）

最后POI l对于用户u的影响因子，可以用下面的等式表示

 （8）

其中 gi是POI l所在的网格。

2.2.3 连续过度影响

这里，我们用POI到POI的过度图来建模签到记录中的连续签到行为，用(l,t)来表示用户已经在时间到t在POI l进行了签到行为，那么POI到POI的过度图可以定义如下。

定义1：用户u的一系列签到行为(l1,t1)，(l2,t2)，…，(ln,tn)，其中t1≦t2≦…≦tn，如果说tn-t1≦τ,我们就说在POI li到li+1 有连续过度影响

定义2：POI到POI的过度图，可以表示为有向图G=(L,E)，其中L是POIs的集合，E是L中所有POIs中有连续过度影响的边。也就是说，如果在所有的用户的历史签到记录中，从li到lj有连续过度影响，那么就存在一条有向边(li,lj)，这个地方，(li,lj)边的权重可以定义为：

 （9）

这里Transitions(li,lj)是所有用户签到记录中，li到lj的连续过度记录的总数。

因为考虑到距离阈值，被推荐给用户的POIs，一定是在用户u的临近网格中，我们可以从全局G中，创建子图G'=(L',E'),从全局G中去除掉不在u临近网格中的POIs，然后再用Edge-weighted Personalized PageRank(EdgePPR)来计算子图中所有结点，对于用户u当前位置的连续转变影响。这个地方选择EdgePPR是因为该算法通过模型降阶，可以在本地高效的运行。用户u到l的连续过度影响可以表示为下面的等式

 （10）

然后利用最大最小归一化方法，对用户偏好，区域影响和连续过度影响进行归一处理



 （11）



这里，，分别是在区域L'中，用户偏好，区域影响，还有连续过度影响的最大最小值，这样对于用户u而言，POI l的得分可以用下面的式子表示

 （12）

其中δ，ε，ξ是满足下面条件的常量

 （13）

3实验设计与验证

4系统的设计和实现

4.1需求分析

4.2系统设计

4.3系统实现

4.4主要界面

4.5本章小结

5总结与展望

5.1总结

两年的研究生生涯，马上就要结束了，在漫漫的求学生涯的最后阶段，有太多需要回味和反思的。这里，我首先要感谢我的导师，刘永坚老师，作为求学生涯的最后一任老师，刘老师以他的大格局深刻影响着我。从计算机出发，以学科融合的态度对待出版行业，在出版行业实现计算机的新型应用，实现出版融合。创造引领行业的新产品。这都是本科阶段，我不曾有过的想法。然后，我也要感谢我的学术指导人，解庆老师，在论文的撰写过程中，我遇到了太多的问题，解老师，以自己丰富的经验和专业的学术功底。给了提了很多建设性建议，无论是从论文的选题开始，还是相关研究的调研，以及后来的论文撰写，解老师都给了我太多的帮助。当然，这里，我还要感谢的我同学们，学术的道路注定是孤独的，在探索真知的过程中，我们都曾迷惘过，但路途中，有了你们的陪伴，再孤独的旅程也会是精彩飞扬。最后，要感谢我的父母。求学二十载，太多的心血放在了我的身上，从牙牙学语，到如今。你们从曾经也在我深夜苦读的旁边陪伴，到后来，我独自踏上求学生涯。无论是生活中，还是学习上，你们都付出了太多，初高中的陪伴，到后来我独自踏上大学，在到如今的研究生生涯。你们一直是我继续下去的理由，感谢有你们。终于到了学业的最后。未来的学习还在继续。

5.2展望

致谢

参考文献

攻读硕士学位期间的研究成果

【1】张荣. O2O模式企业的发展现状[J]. 物流工程与管理, 2013, 35(12):127-129.

【2】. Cheng, C., Yang, H., Lyu, M.R., King, I.: Where you like to go next: Successive point-of-interest recommendation. In: Proceedings of the 23rd International Conference on Artificial Intelligence, pp. 2605–2611 (2013)

【3】He, J., Li, X., Liao, L., Song, D., Cheung, W.K.: Inferring a personalized next point-of-interest recommendation model with latent behavior patterns. In: Proceedings of the 30th AAAI Conference on Artificial Intelligence (2016)

【4】Zhao, S., Zhao, T., Yang, H., Lyu, M.R., King, I.: Stellar: Spatial-temporal latent ranking for successive point-of-interest recommendation. In: Proceedings of the 30th AAAI Conference on Artificial Intelligence (2016)

【5】Feng, S., Li, X., Zeng, Y., Cong, G., Chee, Y.M., Yuan, Q.: Personalized ranking metric embeddingfor next new poi recommendation. In: Proceedings of the 24th International Conference on ArtificialIntelligence, pp. 2069–2075 (2015)

【6】Liu, X., Liu, Y., Li, X.: Exploring the context of locations for personalized location recommendations. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, IJCAI’16, pp. 1188– 1194. AAAI Press (2016)

【7】Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Systems, pp. 3111–3119. Curran Associates Inc. (2013)

【8】Weston, J., Bengio, S., Usunier, N.: Large scale image annotation: Learning to rank with joint wordimage embeddings. In: European Conference on Machine Learning (2010)