DOI:10.13878/j.cnki.jnuist.2019.03.001

连德富[1,2]　谢幸[3]　陈恩红[1,2]

推荐系统中物品召回技术的研究进展

摘要

信息技术的快速发展导致信息过载.推荐系统是解决信息过载最有效的方式之一.近年来,深度学习的快速发展也带动了推荐系统的进步,各种深度推荐算法层出不穷.然而由于候选物品数量巨大且用户兴趣动态变化,深度推荐算法的推荐复杂度巨大,难以在实际系统中单独使用.在深度推荐技术发展的同时,物品召回技术(也称近似搜索技术)也有了较大的发展与进步.本文先介绍基于距离最小化的物品召回的研究进展,再从向量索引、局部敏感哈希、哈希学习、向量量化四个方面来深入探讨基于内积最大化的物品召回技术的研究进展.

关键词

最大内积搜索;召回;最近邻搜索;推荐系统;协同过滤;深度推荐

中图分类号 TP301;TP391

文献标志码 A

收稿日期 2019-05-16

资助项目 国家自然科学基金(U1605251,61832017,61631005,61502077)

作者简介

连德富,男,博士,研究员,博士生导师,主要研究方向为时空数据挖掘、推荐系统、深度学习及其应用.liandefu@ustc.edu.cn

1 中国科学技术大学 计算机科学与技术学院,合肥,230026

2 中国科学技术大学 大数据学院,合肥,230026

3 微软亚洲研究院,北京,100080

0 引言

随着互联网技术的迅猛发展和智能设备的普及,社会进入高度信息化的时代.互联网随处可见,人们的生活方式正在发生巨大变化.随着 Web 2.0 的出现,互联网用户不仅是网络信息的消费者,更是网络内容的生产者.互联网中的信息量呈现爆炸式的增长,比如,淘宝网的在线商品数量高达数十亿,豆瓣上的收录的图书数量高达数百万本.用户在面对庞大复杂的信息时往往面临选择困难,在信息海洋中找到有用信息的成本巨大.这便产生了所谓的"信息过载"问题.

推荐系统是解决信息过载最有效的方式之一.推荐系统通过分析用户的行为历史来对用户兴趣建模,主动地给用户推荐可能符合他们兴趣且满足他们需求的物品.推荐系统的输入数据主要包括用户对物品的评分、用户特征以及物品特征.推荐系统的方法包括基于内容的方法、基于协同过滤的方法以及混合推荐方法.早期的推荐算法主要以矩阵分解模型及其扩展为主.其中典型模型包括 BPR[1]、SVD++[2]、因子分解机(FM)[3]、加权矩阵分解[4]等.在深度学习蓬勃发展之时,深度推荐模型也得以快速进步,主要利用深度学习解决特征建模以及增强高阶特征交互能力.比较典型的方法包括 DSSM[5]、Deep&Wide[6]、DeepCross[7]、NCF[8]、DeepFM[9]、XDeepFM[10]等.深度推荐模型可以提升推荐精度,特别在考虑用户或物品特征时.然而其计算开销巨大,难以为大量的候选物品计算得分并选出高分物品.由于用户兴趣动态变化,实际场景中也难以通过预先计算的方式来实现实时动态推荐.因此,实际推荐系统一般被分为两个阶段,物品召回阶段与细排阶段.在物品召回阶段,利用近似搜索算法帮助用户找到候选物品;在细排阶段,包括深度推荐算法在内的复杂推荐算法对召回物品进行细排推荐.学术界更多关注于复杂推荐算法,较少关注近似搜索算法的设计与改进研究.

本文的主要贡献就在于对近年来推荐系统中近似搜索算法研究进展的概括与总结.我们将近似搜索算法分为四类:基于向量索引的召回技术、基于数据无关哈希的召回技术、基于哈希学习的召回技术以及基于向量量化的召回技术.这些近似搜索算法的核心是将用户和物品分别用向量来表示.所有物品组成待查询集合,用户向量作为查询向量,从而将召回问题转化为检索问题.这类检索问题基于内积、欧式距离或者余弦相似度等来衡量用户对物品的偏好,根据距离最小

化、内积最大化或者相似度最大化来为每个用户检索感兴趣的物品.近年来,也有少数几个基于神经网络逼近的任意度量函数的近似搜索研究[11].然而,内积最大化是推荐系统中物品召回更加常见的检索方式,英文中称为最大内积搜索,简称 MIPS.这可能受以下几个因素影响:第一,内积是双线性函数,梯度计算简单,而余弦相似度涉及向量模长的倒数,可能会面临梯度爆炸的问题,而距离需要开根号,同样可能会面临梯度爆炸的问题;第二,推荐系统中包括矩阵分解在内的很多算法利用内积估计用户偏好,从而基于内积最大化搜索的物品召回可以从中受益,不过最大内积搜索也面临一个重要挑战,即内积并不是一个合法的度量,不满足三角不等式等度量的基本条件,从而近似搜索可能会导致较大的误差.下面先介绍距离最小化、内积最大化以及相似度最大化之间的关系,再介绍多种召回技术.

1 物品召回的计算方式以及关系

假设有 M 个用户,其向量分别用 $\{\boldsymbol{u}_1,\boldsymbol{u}_2,\cdots,\boldsymbol{u}_M\}$ 来表示,向量长度为 d.查询向量用 $\boldsymbol{u}$ 表示.假设有 N 个物品,其向量表示为 $V=\{\boldsymbol{v}_1,\boldsymbol{v}_2,\cdots,\boldsymbol{v}_N\}$.内积最大化搜索通过以下方式来计算:

$$\boldsymbol{v}^* = \arg\max_{\boldsymbol{v}\in V}\langle \boldsymbol{u},\boldsymbol{v}\rangle.$$

余弦相似度最大化的计算方式是:

$$\boldsymbol{v}^* = \arg\max_{\boldsymbol{v}\in V}\frac{\langle \boldsymbol{u},\boldsymbol{v}\rangle}{\|\boldsymbol{v}\|}.$$

最近邻搜索则通过以下计算方式来表达:

$$\boldsymbol{v}^* = \arg\min_{\boldsymbol{v}\in V}\|\boldsymbol{u}-\boldsymbol{v}\|^2 = \arg\max_{\boldsymbol{v}\in V}\langle \boldsymbol{u},\boldsymbol{v}\rangle - \frac{1}{2}\|\boldsymbol{v}\|^2.$$

可以看到这 3 种计算方式存在密切关系.文献[12]中讨论 3 种计算方式的相互变换方法.特别地,如果将用户向量和物品向量分别通过如下方式增广:

$$\hat{\boldsymbol{u}} = [\boldsymbol{u},0],\ \hat{\boldsymbol{v}} = \left[\boldsymbol{v},\sqrt{\max_{\boldsymbol{v}'\in V}\|\boldsymbol{v}'\|^2 - \|\boldsymbol{v}\|^2}\right],$$

那么最大化内积搜索与最近邻查询是等价的,因为

$$\arg\min\|\hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\|^2 = \|\boldsymbol{u}\|^2 - 2\langle \boldsymbol{u},\boldsymbol{v}\rangle + \max_{\boldsymbol{v}'\in V}\|\boldsymbol{v}'\|^2.$$

这种向量增广方式实际上将物品向量嵌入到 $d+1$ 维的半径等于 $\sqrt{\max_{\boldsymbol{v}'\in V}\|\boldsymbol{v}'\|^2}$ 的球面上,因此其空间拓扑结构可能发生了一定的变化.由于对 $\boldsymbol{u}$ 的变换函数和对 $\boldsymbol{v}$ 的变换函数不一致,这种变换被称为非对称变换.然而,正如文献[13]指出的,物品向量最大模长和用户向量模长的差异会导致较大的失真误差.因此,文献[13-14]将查询向量和待查询向量投影到同一个球面上.而且因为作用在 $\boldsymbol{u}$ 上变换函数与 $\boldsymbol{v}$ 保持一致,所以这种变换被文献[14]称为对称变换.文献[13]与文献[14]的差别在于文献[13]保持物品向量的模长不变,而文献[14]则用最大模长归一化了物品向量,使得物品向量在变换之后落到了单位球的球面上.

最大相似度搜索和最近邻查询的变换关系与上述的变换类似,因为给 $\boldsymbol{v}$ 增广之后,所有物品的向量 $\boldsymbol{v}$ 的模长都恒等了.而最大余弦相似度搜索和最大内积搜索的最主要区别就在于后者需要考虑物品向量 $\boldsymbol{v}$ 的模长.

文献[15]提出了另外一种在最大内积搜索和最近邻搜索间建立联系的方法.特别地,$\boldsymbol{u}$ 和 $\boldsymbol{v}$ 通过如下方式增广:

$$\hat{\boldsymbol{u}} = [\boldsymbol{u},\|\boldsymbol{u}\|^2,\|\boldsymbol{u}\|^4,\cdots,\|\boldsymbol{u}\|^{2^m}],$$

$$\hat{\boldsymbol{v}} = \left[\boldsymbol{v},\frac{1}{2},\frac{1}{2},\cdots,\frac{1}{2}\right],$$

那么 $\|\hat{\boldsymbol{u}}-\hat{\boldsymbol{v}}\|^2 = 1+\frac{m}{4}-2\langle \boldsymbol{u},\boldsymbol{v}\rangle + \|\boldsymbol{u}\|^{2^{m+1}}$,所以当 $\boldsymbol{u}$ 的模长小于 1,$\|\boldsymbol{u}\|^{2^{m+1}}$ 就近似为 0,这样便建立了内积和距离的关系.然而,由于增广 m 个元素,而且内积和距离是近似关系而非等价关系,实际往往会采用文献[12]的方法.

2 基于距离最小化的召回技术

由于最近邻查询是基于距离度量的,它在计算机理论和应用领域已经被大量研究,并且一系列或具理论保证或高效快速的近邻查询方法被提出.因此,推荐系统中物品召回的部分研究就是基于最近邻查询的.这类研究主要分成两种,第一种是将用户和物品投影到包括欧式和余弦距离在内的度量空间而非内积空间[16-19].也就是说,用户对物品的偏好是通过距离度量来衡量的,即 $d_{ij} = \|\boldsymbol{u}_i - \boldsymbol{v}_j\|$.文献[16]优化均方误差损失,不过为了提升计算效率,该方法使用距离平方估计偏好.在不考虑偏差的情况下,其优化函数为

$$\sum_{i,j} w_{ij}(r_{ij} - d_{ij}^2)^2 + \lambda d_{ij}^2,$$

其中右边项为正则化,表示用户向量和物品向量不能距离太远.不过损失函数已经不是 $\boldsymbol{u}$ 或者 $\boldsymbol{v}$ 的凸函

数,优化会变得更困难一些.文献[17]优化加权排序损失而且也使用距离的平方来表示用户对物品的偏好,其目标函数为

$$\sum_i \sum_{j\in\Omega_i} \sum_{k\notin\Omega_i} w_{ij}[\epsilon + d_{ij}^2 - d_{ik}^2]_+,$$

其中 $[x]_+ = \max(0,x)$ 是标准的 Hinge 损失,Ω_i 表示用户 i 的正样本集合,而 w_{ij} 是非负权重系数.如果用户到负样本的距离比用户到正样本的距离还大 ϵ,其损失就为 0.ϵ 是 Hinge 损失的间隔.由于使用了分段线性的 Hinge 损失,相比于前面的损失函数,其优化可能会更加容易一些.其实先于文献[17],文献[18]就已经提出了基于排序的优化损失,不过使用了 sigmoid 类型的成对排序损失和直接使用距离来表示用户偏好.然而,这会使得损失的优化较为困难,甚至可能发生梯度爆炸.文献[19]和其他 3 项研究不同,设计了一个基于物品(item-based)的模型,将物品映射到欧式空间中.该方法适用场景是基于用户正在交互的商品来快速搜索出用户可能会购买的商品,背后的动机是利用在购买记录中商品的共现关系.文献[20]基于余弦距离设计目标函数,具体而言,用户对物品的偏好是通过用户向量和物品向量的余弦夹角来刻画的:

$$\theta_{ij} = \arccos \frac{\langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle}{\| \boldsymbol{u}_i \| \| \boldsymbol{v}_j \|},$$

基于该偏好估计,利用成对排序模型来进行学习.如果 j 是正样本,随机采样的负样本为 k,那么其损失为 $-\log \sigma(\theta_{ik} - \theta_{ij})$.

另外一种方案是给基于内积计算的推荐算法添加向量模长恒等的约束,正如文献[21]所提出的.具体而言,其优化目标是:

$$\sum_{(i,j)\in\Omega} (r_{ij} - \langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle)^2, \text{s.t.} \| \boldsymbol{v}_j \| = 1.$$

不过文献[21]并未使用投影梯度下降法,而是通过扩展概率矩阵分解模型,利用后验概率近似技术来进行算法的求解优化.在扩展的概率矩阵分解模型中,冯米塞斯-费舍尔分布作为先验分布被引入刻画常量模长约束.该分布是冯米塞斯的高维扩展,而冯米塞斯是圆上概率分布,也被称为循环正态分布.冯米塞斯-费舍尔分布的密度函数为

$$P(\boldsymbol{v}) = \frac{\exp(\kappa \langle \boldsymbol{\mu}, \boldsymbol{v} \rangle)}{Z_d(\kappa)},$$

其中 $Z_d(\kappa)$ 为归一化常数,$\boldsymbol{\mu}$ 为方向均值,满足 $\|\boldsymbol{\mu}\| = 1$.为了从关于 $\boldsymbol{v}$ 的后验分布中高效地采样向量,该文献的作者提出利用短程蒙特卡罗(Geodesic Monte Carlo)方法.

在学习得到距离度量空间中的用户与物品向量之后,便可以采用最近邻搜索来召回物品.关于最近邻搜索方法不在本文中阐述,具体可以参考文献[22].文献[22]在多个大数据集上对很多最近邻搜索方法进行了系统性的对比.一个有趣的数据结构 PCA-Tree 值得在这里讨论,因为该数据结构是针对推荐问题中的最大内积搜索提出的[12],却适用于普通的最近邻搜索.具体而言,将带查询向量组织成一个矩阵 $\boldsymbol{V}$,用截断奇异值分解得到其 d' 个最大的左奇异向量,用这 d' 个左奇异向量同时旋转用户向量和物品向量,用 KD-Tree 对旋转后的向量进行索引.这个方法的主要好处在于 $d' < d$,而且是 d' 个最重要的维度,因而可以用在低维情形表现更好的 KD-Tree 直接索引.

3 基于内积最大化的召回技术

上一节介绍了关于基于距离最小化的召回技术的研究进展,其主要思路是设计适合于最近邻搜索的目标函数或者约束条件.不过这些研究无法从已有的大量基于内积计算的推荐算法中受益.本节将介绍直接基于内积计算的物品召回.

3.1 基于向量索引的召回技术

第一类方法是基于索引的技术,也就是为最大内积搜索建立索引,然后再基于索引进行搜索.较早的研究文献首先通过递归方式构建了度量树(Metric Tree)[23].度量树是二叉空间划分树,将空间划分为可重叠的球体.一种简单的度量树的构造方法是,每次划分向量点集时,先找到两个距离尽量远的向量作为初始中心点,然后将该点集中的所有向量划分到最近的中心点上,并用向量均值来更新中心点.点集是通过递归方式进行划分的,直到待划分的向量点集大小小于设定的阈值.在度量树构建完成之后,便可以利用深度优先搜索进行物品搜索.不过,由于内积不满足三角不等式,搜索剪枝必须依赖新的不等式,否则无法实现搜索加速.文献[23]提出了内积计算的新上界用于搜索剪枝:

$$\sum_{v_j\in B_{v_c}^r} \langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle \leqslant \langle \boldsymbol{u}_i, \boldsymbol{v}_c \rangle + r \| \boldsymbol{u}_i \|,$$

其中 $B_{v_c}^r$ 是以 $\boldsymbol{v}_c$ 为中心的,半径为 r 的包含物体向量的球.在深度优先搜索时,一般维持一个优先队列,保存遍历时已经找到的最大内积的物品及其内积值.当遍历到树中某个节点时,如果用户对节点里物

品的最大内积上界小于优先队列中的最小内积值,那么这个节点便可以直接剪掉,不必进一步探索.

为了提升查询效率,度量树在每次划分时尽量保证左右子树的物品数要尽量相等,正如在文献[11]提到的和工业开源查询算法库 Annoy(https://github.com/spotify/annoy)实现的那样.有趣的是,文献[11]并没有使用树节点的向量均值来作为树节点的向量表示,而是通过推荐模型来学习.不过构造的树不再是度量树,无法用新上界来进行剪枝加速.具体而言,如果树节点包含了用户 i 正样本的物品向量,那么该节点作为用户 i 在对应层的正样本,同一层中不包含用户 i 正样本的树节点作为负样本,从而可以通过优化方法来学习树节点的向量表示.需要注意的是,为了保证层对齐,该文献假设每个叶子节点只含有一个物品向量.在该树上的搜索是通过宽度优先搜索完成的,搜索每一层时,用优先队列保留偏好分值最大的 k 节点,k 为召回的物品数.搜索到最后一层时,优先队列保存的节点便是召回的 k 个物品.这种启发式的搜索实际上是一种近似算法.

最后介绍一种基于向量模长的高效召回算法[24-25].该方法的主要思路是根据向量模长对物品向量分组,然后利用柯西施瓦茨不等式 $\langle \boldsymbol{u},\boldsymbol{v}\rangle \leqslant \|\boldsymbol{u}\|\ \|\boldsymbol{v}\|$ 进行剪枝.其中物品向量的分组类似于上面提到的物品向量聚类.当然文献[24-25]还设计了更加复杂的剪枝策略,比如基于夹角的剪枝、增量式的剪枝等.

除了在物品向量上建索引之外,往往还可以对用户向量聚类,用聚类中心表示用户.因为只要将少数的聚类中心作为查询向量,所以可以进一步地提升搜索效率.这类方法在文献[23,26]中有提及,甚至还为用户聚类中心设计了内积上界,以用于搜索剪枝.

3.2 基于数据无关哈希的召回技术

数据无关哈希指局部敏感哈希,主要因为其哈希函数和数据无关.局部敏感哈希的最大特点是哈希值保持数据的相似性,即越相似数据的哈希值越相近.哈希函数族对相似度函数 $\mathrm{sim}:(U,V)\to(0,1)$ 是 (S,cS,p_1,p_2) - 敏感的,如果满足:

● 若 $\mathrm{sim}(\boldsymbol{u},\boldsymbol{v})\geqslant S$,那么 $\mathbb{P}_h[h(\boldsymbol{u})=h(\boldsymbol{v})]\geqslant p_1$,

● 若 $\mathrm{sim}(\boldsymbol{u},\boldsymbol{v})\leqslant cS$,那么 $\mathbb{P}_h[h(\boldsymbol{u})=h(\boldsymbol{v})]\leqslant p_2$.

一般要求 $p_1>p_2$ 且 $c<1$.该方法的重要理论结果是,(S,cS,p_1,p_2) -敏感哈希可以在 $O(n^{\rho}\log n)$ 时间内构造占 $O(n^{1+\rho})$ 空间的数据结构用于最近邻查询[27],其中 $\rho=\dfrac{\log p_1}{\log p_2}$.可以看到该哈希函数族是对称的,即作用在 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 上的哈希函数是相同的.与之相关的非对称哈希,是指在 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 上作用的哈希函数不同.在前面讨论的有关内积最大化到距离最小化的关系中,文献[15]对 $\boldsymbol{u}$ 的变换 P 和对 $\boldsymbol{v}$ 的变换 Q 是不同的,尽管变换之后应用的哈希函数 h 相同,但是作用在 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 上的哈希函数分别是复合函数 $h\circ P$ 和复合函数 $h\circ Q$.哈希函数族对相似度函数 $\mathrm{sim}:(U,V)\to(0,1)$ 是 (S,cS,p_1,p_2) - 非对称敏感,如果满足:

● 若 $\mathrm{sim}(\boldsymbol{u},\boldsymbol{v})\geqslant S$,那么 $\mathbb{P}_{(f,g)}[f(\boldsymbol{u})=g(\boldsymbol{v})]\geqslant p_1$,

● 若 $\mathrm{sim}(\boldsymbol{u},\boldsymbol{v})\leqslant cS$,那么 $\mathbb{P}_{(f,g)}[f(\boldsymbol{u})=g(\boldsymbol{v})]\leqslant p_2$.

根据文献[14],在实数域上,不存在普适的对称函数族和非对称哈希函数族.若 U 在单位球上,即 $\|\boldsymbol{u}\|=1$,V 在单位球内,即 $\|\boldsymbol{v}\|\leqslant 1$,则存在普适的对称哈希函数族.其变换函数是一样的,即:

$$P(\boldsymbol{u})=\left[\boldsymbol{u},\sqrt{\max_{\boldsymbol{u}'\in U}\|\boldsymbol{u}'\|-\|\boldsymbol{u}\|}\right]=\left[\boldsymbol{u},0\right],$$

$$Q(\boldsymbol{v})=\left[\boldsymbol{v},\sqrt{\max_{\boldsymbol{v}'\in V}\|\boldsymbol{v}'\|-\|\boldsymbol{v}\|}\right]=\left[\boldsymbol{v},\sqrt{1-\|\boldsymbol{v}\|}\right].$$

若 U 和 V 均为单位球时,即 $0\leqslant\|\boldsymbol{u}\|,\|\boldsymbol{v}\|\leqslant 1$,则不存在普适的对称哈希函数族,但存在普适的非对称哈希函数族,其变换分别为

$$P(\boldsymbol{u})=\left[\boldsymbol{u},\sqrt{1-\|\boldsymbol{u}\|},0\right],$$

$$Q(\boldsymbol{v})=\left[\boldsymbol{v},0,\sqrt{1-\|\boldsymbol{v}\|}\right].$$

不过由于在用内积计算时,查询向量的模长并不会影响最大内积搜索的结果,因此可以利用对称哈希.这要求算法对数据做预处理,即用户向量归一化使得长度为 1,而物品向量则除以所有物品向量模长的最大值.

在变换之后,哈希函数 h 的选择有两类方案.一种是基于距离的哈希函数,记为 L2-LSH,定义为

$$h^{L2}_{a,b}(\boldsymbol{u})=\left\lfloor\frac{\langle\boldsymbol{a},\boldsymbol{u}\rangle+b}{r}\right\rfloor$$

其中 $\boldsymbol{a}\sim\mathcal{N}(0,\boldsymbol{I})$,$b\sim U(0,r)$,$r$ 为预设的参数.在这种情况下,冲突的概率为

$$\mathrm{P}\left(h^{L2}_{a,b}\left(\boldsymbol{u}\right)=h^{L2}_{a,b}(\boldsymbol{v})\right)=$$

$$1-2\Phi(-r/d)-\frac{2}{\sqrt{2\pi}(r/d)}\left(1-\exp\left\{-\frac{1}{2}(r/d)^2\right\}\right),$$

其中 $d = \|\boldsymbol{u}-\boldsymbol{v}\|$,$\Phi(x)$ 为标准正态分布的累积密度函数.所以,距离越小,冲突的概率越大,即越可能是同一个哈希值.在实际应用中,一般会生成长度为 d 的哈希码,然后计算哈希码匹配的比例用作检索的依据[15].

另外一种方式是距离相似度的哈希函数,记为 SIGN-LSH,定义为

$$h^{\text{sign}}(\boldsymbol{u}) = \text{sign}(\langle \boldsymbol{a},\boldsymbol{u}\rangle),$$

其中 $\boldsymbol{a} \sim N(0,\boldsymbol{I})$.这种方法也称为随机投影符号法.在这种情况下,冲突的概率为

$$\text{P}\left(h^{\text{sign}}(\boldsymbol{u}) = h^{\text{sign}}(\boldsymbol{v})\right) = 1-\frac{1}{\pi}\cos^{-1}\left(\frac{\langle \boldsymbol{u},\boldsymbol{v}\rangle}{\|\boldsymbol{u}\|\,\|\boldsymbol{v}\|}\right),$$

那么 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 越相似,冲突的概率越大,即越有可能是相同的哈希值.在实际应用中,一般也会生成长度为 d 的哈希码,然后计算用户向量和物品向量的哈希码的汉明距离作为检索的依据[14].

不过由于物品向量的模长一般满足长尾分布,在用最大模长归一化之后,很多物品向量的模长很小,使得近似查找性能受到很大的影响.为此,文献[13,28]提出将物品向量根据向量模长进行分组,然后在组内建立哈希索引.通过理论和实践证明,这种方法能显著地提升检索的性能.

除了给定用户向量和物品向量的局部敏感哈希算法以外,还有针对新闻等文本的局部敏感哈希[29],即用 Jaccard 距离来度量集合之间的相似度.这个局部敏感哈希算法被称为 MinHash.文献[29]利用 MinHash 检索相似的新闻以实现快速的新闻推荐.

3.3 基于哈希学习的召回技术

从上节可以看到,哈希函数的参数是从正态分布中采样得到的,和数据是无关的.因此,局部敏感哈希也称为数据无关哈希.这种方法产生的哈希码往往低效,只有当哈希码足够长时,才能有足够的近似精度[30],而且这种近似精度随码长递增的趋势非常缓慢.因此,先驱性的研究工作提出了从数据中学习哈希函数,希望哈希函数学到的哈希码之间的汉明距离能逼近数据向量之间的距离[30-31].换句话说,即相似的物品应具有相似的哈希码.更具体地,给定向量数据点集 $\{\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N\}$,文献[31]提出利用堆叠的受限布尔兹曼机学习每个数据点的隐层表示.从输入到隐层的非线性映射函数,即编码器,就是哈希函数.通过这种方式学习得到的哈希码在效率和准确度上都显著优于数据无关哈希.文献[30]研究哈希码的最优特性:$\sum_i \boldsymbol{y}_i = 0$,使得每个比特上取 1 或者-1 的概率尽可能相同;$\frac{1}{N}\sum_i \boldsymbol{y}_i\boldsymbol{y}_i^{\text{T}} = \boldsymbol{I}$,使得不同比特位尽可能独立.在这两个约束下,最小化相似数据点的平均汉明距离:

$$\min_{\boldsymbol{y}_i,\boldsymbol{y}_j\in\{\pm 1\}^d}\sum_{i,j} w_{ij}\|\boldsymbol{y}_i-\boldsymbol{y}_j\|^2,$$

其中 $w_{ij} = \exp(-\|\boldsymbol{x}_i-\boldsymbol{x}_j\|^2/\epsilon^2)$ 表示数据点 i 和数据点 j 的相似度.若码长为 1,该目标函数的优化对应平衡图分割问题,是 NP 难问题.因此,该文献通过松弛法后计算特征值分解来求解.

这些一般非推荐搜索场景中的哈希算法研究成果非常丰富,该节不做详细介绍,具体可以参考调研文献[32].在将这些算法应用到推荐系统中的物品召回时,可以将增广后的用户向量和物品向量同时放入数据点集,同时学习用户和物品的哈希码.然而,这种解决方案无法考虑到推荐算法自身的特点.因此,该节关注针对推荐算法设计的哈希学习方法.较早的算法是文献[33]提出的,包含两个方面的显著差异.第一,修改目标函数以便得到的用户向量和物品向量能生成更高效的哈希码,即优化以下目标函数:

$$\sum_{i,j} w_{ij}(r_{ij}-\langle \boldsymbol{u}_i,\boldsymbol{v}_j\rangle)^2 + \lambda\left(\left\|\sum_i \boldsymbol{u}_i\right\|^2 + \left\|\sum_j \boldsymbol{v}_j\right\|^2\right).$$

正则化约束用户向量的平均值和物品向量的平均值接近 0,使得根据符号计算哈希值可以满足最优哈希码的第一个特性.第二,同时对 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 做相同的旋转变换,内积值不会改变,但 $\boldsymbol{u}$ 和 $\boldsymbol{v}$ 的旋转可以帮助解决不同维度信息量存在显著差异的问题.不同维度的信息量差异对生成哈希码质量的影响在文献[34-35]中被仔细研究,为此,他们提出了空间旋转法这一种重要方法.在文献[33]中,作者提出如下的目标函数来从用户向量和物品向量中学习各自的哈希码:

$$\min_{\boldsymbol{s}_i,\boldsymbol{t}_j\in\{\pm 1\}^d,\boldsymbol{W}}\sum_i \|\boldsymbol{s}_i-\boldsymbol{W}\boldsymbol{u}_i\| + \sum_j \|\boldsymbol{t}_j-\boldsymbol{W}\boldsymbol{v}_j\|,$$

$$\text{s.t.}\sum_i \boldsymbol{s}_i = 0,\ \sum_j \boldsymbol{t}_i = 0,$$

$\boldsymbol{W}$ 为正交矩阵.

从用户和物品向量直接生成哈希码最重要的问题是用户向量和物品向量的模长丢失.文献[36]则提出模长恒等正则来约束用户向量和物品向量的学习.特别地,文献[36]提出优化如下的目标函数来学习用户向量和物品向量:

$$\sum_{i,j} w_{ij}(r_{ij} - \langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle)^2 + \lambda \sum_i (\| \boldsymbol{u} \|_i - c)^2 + \lambda \sum_j (\| \boldsymbol{v}_j \| - c)^2,$$

其中c设为最大分值的一半,主要是为了匹配内积值和评分值各自的范围.由于这个正则化无法约束模型直接获得模长恒等的用户向量和物品向量,直接量化仍然会丢失模长信息.为了进一步降低模长丢失带来的损失,作者进一步提出了模长量化的方法,使得余弦相似度和模长被同时量化,从而提升哈希码的表达能力.

上述的两个方法的共性是先从评分数据中学习用户和物品向量,然后再从这些向量中学习对应的哈希码.这类方法统称为两阶段法.由于学习得到的用户和物品向量可能不是最适合用来生成哈希码的,这种两阶段法可能会有较大的量化损失.为此,文献[37]提出从评分数据中学习高效哈希码的方法,即优化如下的目标函数:

$$\sum_{i,j} w_{ij}(r_{ij} - \langle \boldsymbol{s}_i, \boldsymbol{t}_j \rangle)^2, \quad \text{s.t.} \sum_i \boldsymbol{s}_i = 0, \sum_j \boldsymbol{t}_j = 0, \frac{1}{M}\sum_i \boldsymbol{s}_i \boldsymbol{s}_i^{\mathrm{T}} = I, \quad \frac{1}{N}\sum_j \boldsymbol{t}_j \boldsymbol{t}_j^{\mathrm{T}} = I.$$

由于涉及到离散变量和约束条件,这个目标函数的优化是非常困难的.为此,作者提出了松弛法,为二值变量引入代理实值变量,然后将约束作用在代理变量上,同时约束二值变量和代理实值变量的差异.此时代理实值变量具有最优的解析解,而二值变量的优化仍然非常困难.文献[37]利用离散坐标下降法,每次只优化一个比特.这种算法在组合优化问题中被称为局部搜索,而且搜索的参数空间包含以当前参数为中心,半径为1-汉明距离的所有参数.为了进一步扩大参数搜索范围,同时保证搜索的效率,文献[38]提出利用半正定松弛法来解决.具体而言,上述问题很容易转化为二值二次规划(BQP)问题的求解:

$$\min_{\boldsymbol{x} \in \{\pm 1\}^d} \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x} - 2\boldsymbol{b}^{\mathrm{T}} \boldsymbol{x}.$$

这个问题会先转化为齐次二值二次规划,即:

$$\min_{[\boldsymbol{x},t] \in \{\pm 1\}^{d+1}} [\boldsymbol{x}^{\mathrm{T}}, t] \begin{bmatrix} \boldsymbol{A} & -\boldsymbol{b} \\ -\boldsymbol{b}^{\mathrm{T}} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{t} \end{bmatrix}.$$

假设$(\boldsymbol{x}^*, t^*)$是齐次解,那么非齐次解为$\boldsymbol{x}^* t^*$.对于齐次二值二次规划,可以有如下的等价关系:

$$\min_{\boldsymbol{x}} \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x}, \ \text{s.t.} \ \boldsymbol{x} \in \{\pm 1\}^d \Leftrightarrow \max_{\boldsymbol{X}} \operatorname{trace}(\boldsymbol{A}\boldsymbol{X}), \ \text{s.t.} \ \operatorname{diag}(\boldsymbol{X}) = 1, \operatorname{rank}(\boldsymbol{X}) = 1.$$

为了求解后者的问题,通常会先把秩约束去除,求解半正定规划问题.在得到半正定规划的最优解$\boldsymbol{X}^*$后,通过从正态分布$\mathcal{N}(0, \boldsymbol{X}^*)$中采样后再应用符号函数就得到哈希码.因此在参数搜索时,汉明距离变大了,但依概率方式进行近似搜索,得以提升参数优化的效率.

然而这些二值优化的推荐算法是直接优化评分误差的,与推荐的目标-物品推荐不一致.为此,文献[39]提出了优化成对排序损失的方法;文献[40]引入了适用于物品推荐的隐性正则.由于引入隐性正则,文献[40]能同时处理评分数据或者单类的隐式反馈数据.另外,该文献还通过变分法将损失函数从评分损失扩展到非线性凸损失,可以处理单类的隐式反馈数据或者正负反馈数据.

上述基于矩阵分解的推荐算法,难以与快速发展的深度推荐算法耦合.然而在深度学习框架中端到端地学习二值的用户向量和物品向量是困难的,因为无法通过后向传播求解梯度.假设物品j的某个二值码$t_j \in \{0,1\}$是服从参数为α_j伯努利分布的随机变量,即$P(t_j = 1) = \dfrac{\alpha_j}{1 + \alpha_j}$.损失函数$E_{t_j \sim P_{\alpha_j}(t_j)}[\ell(t_j)]$对$t_j$的梯度无法回传到$\alpha_j$上.文献[41]利用一种新提出的离散分布[42-43]来近似伯努利分布的随机变量,即:

$$x_j = \frac{1}{1 + \exp(-(\log \alpha_j + \log \epsilon - \log(1 - \epsilon))/T)},$$

其中$\epsilon \sim U(0,1)$.随机变量x_j的两个重要性质是$P(x_j > 0) = \dfrac{\alpha_j}{1 + \alpha_j}$;$P(\lim\limits_{T \to 0} x_j = 1) = \dfrac{\alpha_j}{1 + \alpha_j}$.也就是说当温度$T$趋于0时,该随机变量服从伯努利分布.更重要的一个性质是参数和采样被分离,所以损失$E_{\epsilon \sim U(0,1)}[\ell(g(\alpha_j, \epsilon))]$对$x_j = g(\alpha_j, \epsilon)$的梯度能回传到$\alpha_j$上.关于二值变量甚至离散变量优化问题,文献[43]提到了直接估计梯度的两类方法.一类是直通估计(Straight-Through estimator)[44],比如梯度$\nabla_{\alpha_j} t_j = 1$.文献[45]为每个二值变量引入连续变量,再通过把二值变量边缘化的方法来避免二值变量的梯度计算.另外一类是REINFORCE估计,最早用于强化学习中的策略梯度估计.该方法的关键是

$$\nabla_{\alpha_j} E_{P_{\alpha_j}}[\ell(t_j)] = E_{P_{\alpha_j}}[\ell(t_j) \nabla_{\alpha_j} \log P_{\alpha_j}(t_j)].$$

不过由于该方法估计的梯度的方差过大,使得算法训练容易震荡,收敛速度缓慢.因而有很多新方法来降低梯度估计的方差,具体可以阅读文献[43]

了解更多信息.

除此以外,还有一种学习二值变量的思路就是近似符号函数.文献[41]通过引入温度 T 到高斯误差函数 $\mathrm{erf}(x)=\frac{2}{\sqrt{\pi}}\int_0^{}\mathrm{e}^{-t^2}\mathrm{d}t$.当温度 T 趋于 0 时,误差函数也逼近符号函数.也可以引入温度 T 到 sigmoid 函数,能得到类似的结论,文献[46]便使用这种方法.这种函数逼近的方法的主要问题是梯度消失.因为当 $|x|$ 大到一定程度之后,x 的显著变化只能导致误差函数或者 sigmoid 函数的细微变化,也即梯度接近 0.实际中一般的策略是,初始时设置较大的温度,随着迭代的进行,逐渐地降低温度直到到达一个较小的温度.

3.4 基于向量量化的召回技术

本节将要介绍的方法是基于向量量化的方法.这个方法和前面提到的度量树有一定的关系,同样需要对物品向量进行聚类.不过度量树是进行层次聚类,每次聚类时是把当前类中的向量一分为二.而向量量化也一般会进行多次聚类,不过每次聚类时都是把所有的物品向量聚成同样个数的类.因此在向量量化时,不同次聚类的差异性就非常重要.文献[47]提出的乘积量化是向量量化的先驱工作.这里的乘积是指笛卡尔乘积.乘积量化先将维度为 d 的向量空间划分为多个维度相同且不相交的子空间,然后在各个子空间上聚成 K 个类.每个物品向量用子空间的中心点拼接而成的向量来近似.特别地,假设用户向量切分为 $\boldsymbol{u}=[\boldsymbol{u}^{(1)},\boldsymbol{u}^{(2)},\cdots,\boldsymbol{u}^{(F)}]$,物品向量切分为 $\boldsymbol{v}=[\boldsymbol{v}^{(1)},\boldsymbol{v}^{(2)},\cdots,\boldsymbol{v}^{(F)}]$.对物品向量分别在对应子空间聚成 K 类,中心点用矩阵 $\boldsymbol{C}^f, f=\{1,\cdots,F\}$ 来表示,而向量和中心点的对应关系用 one-hot 向量 $\boldsymbol{\alpha}^{(f)}$ 来表示.那么 $\boldsymbol{v}$ 就是用 $\sum_f \boldsymbol{C}^{(f)}\boldsymbol{\alpha}_v^{(f)}$ 表示.那么内积计算可以通过如下方式来表达:

$$\langle \boldsymbol{u},\boldsymbol{v}\rangle=\sum_f\left\langle \boldsymbol{u}^{(f)},\boldsymbol{C}^{(f)}\boldsymbol{\alpha}_v^{(f)}\right\rangle=\sum_f\left\langle \boldsymbol{u}^{(f)},\boldsymbol{c}_v^{(f)}\right\rangle,$$

其中 $\boldsymbol{c}_v^{(f)}$ 表示向量 $\boldsymbol{v}$ 在子空间 f 中对应的中心点.由于只需要和中心点计算内积,内积的计算可以被加速.由于不同子空间的划分能导致不同的乘积量化结果,自然需要寻求最优子空间划分.为此,文献[48-49]提出了联合优化的方法,一方面寻找最优子空间划分,一方面做乘积量化.这里的最优子空间划分实际上是通过对空间的旋转来实现的.注意到,维度重排实际上也对应空间旋转.空间旋转能解决子空间最优划分的原因实际上可以用文献[34]的动机来解释.即在一般情况下各个维度所蕴含的信息量不同,因此在不同信息量的子空间下用相同数量的聚类来表达,信息丢失是较大的.理想情况是要么信息量大的子空间应该用更多类的聚类来表示,要么不同子空间的信息量接近相同.

除了上述的子空间聚类,还有一些在全空间上的聚类方法,只要能使得不同次聚类的结果有较大差异即可.比如文献[50]使用残差向量进行聚类.第 1 次,使用聚类方法对所有的向量聚类.对于向量 $\boldsymbol{x}$,第 1 次聚类之后用中心点表示 $\tilde{\boldsymbol{x}}_1$,残差为 $\boldsymbol{\epsilon}_1=\boldsymbol{x}-\tilde{\boldsymbol{x}}_1$;第 f 次在残差向量 $\boldsymbol{\epsilon}_f$ 上聚类,用 $\tilde{\boldsymbol{x}}_f$ 表示对于中心点,残差为 $\boldsymbol{\epsilon}_f=\boldsymbol{x}-\sum_{f'\leqslant f}x_{f'}$;直到第 F 次聚类完成.文献[51-52]提出了加性量化的方法,旨在解决残差向量量化中前 f 次聚类不受 $f+1$ 到 F 次聚类影响的问题.文献[52]用束搜索(beam search)的方法来进行中心点编码(即属于哪个中心点),而用最小二乘的方法来学习中心点.在束搜索方法中,首先从 $F\times K$ 个中心点找到 N 个离 $\boldsymbol{x}$ 最近的中心点.这 N 个中心点表示在只有 1 次聚类时最好的 N 个近似.在 $f(f\geqslant 2)$ 次迭代时,假设已经有 $f-1$ 次聚类时最好的 N 个近似.然后针对每个近似,从其余的 $(M-f+1)\times K$ 个中心点中找到 $\boldsymbol{x}$ 与该近似的差值最接近的 N 个中心点,总共得到 f 次聚类时的 N^2 个结果,再从中找到 N 个互异且最好的 N 个近似.$F-1$ 次迭代之后,便能找到较优的中心点编码.文献[51]则使用 Iterated Conditional Modes 进行中心点编码.具体而言,就是针对输入 $\boldsymbol{x}$,先固定 $\boldsymbol{x}$ 的 $F-1$ 个中心点编码,再学习剩下的一个中心点编码,即在剩下的 K 个中心点中找到最优的近似.这个过程迭代进行直到收敛.可以看到,文献[51]同样解决了残差向量量化中前 f 次聚类不受 $f+1$ 到 F 次聚类影响的问题.针对上述的两个模型,还有很多的改进算法,比如引入树结构的量化[53]以及加性量化的编码加速[54-55].

为了将这些向量量化用于近似的最大内积搜索,一方面需要利用前面提到的内积最大化和距离最小化之间的关系进行向量变换;另一方面则需要建立类似于度量树的索引,比如文献[47]基于 K-means 聚类建立了基于倒排表的索引,再结合基于输入向量与聚类中心差值的向量量化来实现近似搜索.该方法的另外一个优势是用残差来做向量量化能更加准确,因为第 1 次的聚类能捕获主要信息,残差表示主要信息之外的细节信息.

除了这些基于距离最小化的向量量化,也有基于内积最大化的向量量化[56].该文献提出两种策略来学习向量的量化.第1种方法是,最小化向量量化前后的内积差异:

$$\min_{C,\alpha}\sum_{i,j}\left(\langle \boldsymbol{u}_i,\boldsymbol{v}_j\rangle-\sum_f\left\langle \boldsymbol{u}_i^{(f)},\boldsymbol{C}^{(f)}\boldsymbol{\alpha}_{v_j}^{(f)}\right\rangle\right)^2.$$

第2种是在最小化内积差异的情况下,再约束物品的相对序在向量量化前后的一致性.针对某个用户,特别考虑了量化前内积最大的物品与其他物品的相对序在量化之后也要保持一致.为了加速优化,无法将所有的其他物品考虑在内,而是从相对序不一致的其他物品中采样.

4 结束语与展望

本文介绍了基于最近邻搜索的物品召回,再从向量索引、局部敏感哈希、哈希学习、向量量化四个方面详细介绍了内积最大化的物品召回.目前有包括Faiss、Annoy、NMSLIB在内的开源库支持推荐系统中的物品召回(https://www.benfrederickson.com/approximate-nearest-neighbours-for-recommender-systems/).

笔者认为以下研究方向需要进一步探索:

1)目前的最好的局部敏感哈希方法是根据向量模长进行物品集合划分后在各个集合上做局部敏感哈希.这种方法是较为经验性的,因而统一的理论框架就显得非常重要.

2)哈希学习中二值码的学习优化方法均可认为是局部搜索算法,而且搜索空间非常有限.比如在深度学习中,每个比特的哈希函数是独立的,并没有考虑比特之间的组合计算.

3)向量量化方法目前主要基于距离度量,而非内积.而且基本是两阶段法或多阶段法,缺少端到端的算法设计,也缺少结合深度推荐算法的向量量化算法.

参考文献

References

[1] Rendle S, Freudenthaler C, Gantner Z, et al. BPR: bayesian personalized ranking from implicit feedback[C]//Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence. AUAI Press, 2009: 452-461

[2] Koren, Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C]//Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2008: 426-434

[3] Rendle S. Factorization machines with libFM[J]. ACM Transactions on Intelligent Systems and Technology, 2012,3(3):1-22

[4] Hu Y F, Koren Y, Volinsky C. Collaborative filtering for implicit feedback datasets[C]//2008 Eighth IEEE International Conference on Data Mining, 15-19 Dec.2008, Pisa, Italy, 2008:263-272

[5] Huang P S, He X, Gao J, et al. Learning deep structured semantic models for web search using clickthrough data [C]//Proceedings of the 22nd ACM International Conference on Information & Knowledge Management. ACM, 2013:2333-2338

[6] Cheng H T, Koc L, Harmsen J, et al. Wide & deep learning for recommender systems[C]//Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. ACM, 2016:7-10

[7] Shan Y, Hoens T R, Jiao J, et al. Deep crossing: web-scale modeling without manually crafted combinatorial features [C]//Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2016:255-262

[8] He X, Liao L, Zhang H, et al. Neural collaborative filtering[C]//Proceedings of the 26th International Conference on World Wide Web, 2017:173-182

[9] Guo H, Tang R, Ye Y, et al. DeepFM: a factorization-machine based neural network for CTR prediction[C]//Proceedings of the 26th International Joint Conference on Artificial Intelligence. AAAI Press, 2017:1725-1731

[10] Lian J, Zhou X, Zhang F, et al. xDeepFM: combining explicit and implicit feature interactions for recommender systems[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2018:1754-1763

[11] Zhu H, Li X, Zhang P, et al. Learning tree-based deep model for recommender systems[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2018: 1079-1088

[12] Bachrach Y, Finkelstein Y, Gilad-Bachrach R, et al. Speeding up the xbox recommender system using a euclidean transformation for inner-product spaces[C]//Proceedings of the 8th ACM Conference on Recommender Systems. ACM, 2014:257-264

[13] Huang Q, Ma G, Feng J, et al. Accurate and fast asymmetric locality-sensitive Hashing scheme for maximum inner product search[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2018: 1561-1570

[14] Neyshabur B, Srebro N. On symmetric and asymmetric LSHs for inner product search[C]//International Conference on Machine Learning, 2015:1926-1934

[15] Shrivastava A, Li P. Asymmetric LSH (ALSH) for sublinear time maximum inner product Search (MIPS)[J]. Advances in Neural Information Processing Systems, 2014,3:2321-2329

[16] Khoshneshin M, Street W N. Collaborative filtering via eu-

clidean embedding[C]//Proceedings of the Fourth ACM conference on Recommender Systems. ACM,2010:87-94
[17] Hsieh C K, Yang L, Cui Y, et al. Collaborative metric learning[C]//Proceedings of the 26th International Conference on World Wide Web,2017:193-201
[18] Le D D, Lauw H W. Euclidean co-embedding of ordinal data for multi-type visualization[C]//Proceedings of the 2016 SIAM International Conference on Data Mining, 2016:396-404
[19] Koenigstein N, Koren Y. Towards scalable and accurate item-oriented recommendations[C]//Proceedings of the 7th ACM Conference on Recommender Systems. ACM, 2013:419-422
[20] Le D D, Lauw H W. Indexable bayesian personalized ranking for efficient top-k recommendation[C]//Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. ACM,2017:1389-1398
[21] Fraccaro M, Paquet U, Winther O. Indexable probabilistic matrix factorization for maximum inner product search [C]//Thirtieth AAAI Conference on Artificial Intelligence,2016
[22] Li W, Zhang Y, Sun Y F, etal. Approximate nearest neighbor search on high dimensional data-experiments, analyses, and improvement[J]. IEEE Transactions on Knowledge and Data Engineering,2019:1
[23] Koenigstein N, Ram P, Shavitt Y. Efficient retrieval of recommendations in a matrix factorization framework[C]//Proceedings of the 21st ACM International Conference on Information and Knowledge Management. ACM,2012: 535-544
[24] Teflioudi C, Gemulla R, Mykytiuk O. Lemp: fast retrieval of large entries in a matrix product[C]//Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. ACM,2015:107-122
[25] Li H, Chan T N, Yiu M L, et al. FEXIPRO: fast and exact inner product retrieval in recommender systems[C]//Proceedings of the 2017 ACM International Conference on Management of Data. ACM,2017:835-850
[26] Ram P, Gray A G. Maximum inner-product search using cone trees[C]//Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM,2012:931-939
[27] Indyk P, Motwani R. Approximate nearest neighbors: towards removing the curse of dimensionality[C]//Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing. ACM,1998:604-613
[28] Yan X, Li J, Dai X, et al. Norm-ranging LSH for maximum inner product search[C]//Advances in Neural Information Processing Systems,2018:2952-2961
[29] Das A S, Datar M, Garg A, et al. Google news personalization: scalable online collaborative filtering[C]//Proceedings of the 16th International Conference on World Wide Web. ACM,2007:271-280
[30] Weiss Y, Torralba A, Fergus R. Spectral hashing[C]//Advances in Neural Information Orocessing Systems, 2009:1753-1760
[31] Salakhutdinov R, Hinton G. Semantic hashing[J]. International Journal of Approximate Reasoning,2009,50(7): 969-978
[32] Wang J D, Zhang T, Song J K, et al. A survey on learning to hash[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence,2018,40(4):769-790
[33] Zhou K, Zha H. Learning binary codes for collaborative filtering[C]//Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM,2012:498-506
[34] Gong Y C, Lazebnik S, Gordo A, et al. Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence,2013,35(12): 2916-2929
[35] Kong W, Li W J. Isotropic hashing[C]//Advances in Neural Information Processing Systems,2012:1646-1654
[36] Zhang Z, Wang Q, Ruan L, et al. Preference preserving hashing for efficient recommendation[C]//Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval. ACM, 2014:183-192
[37] Zhang H, Shen F, Liu W, et al. Discrete collaborative filtering[C]//Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM,2016:325-334
[38] Zhang Y, Wang H, Lian D, et al. Discrete ranking-based matrix factorization with self-paced learning[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2018:2758-2767
[39] Zhang Y, Lian D, Yang G. Discrete personalized ranking for fast collaborative filtering from implicit feedback[C]// Thirty-First AAAI Conference on Artificial Intelligence,2017
[40] Lian D, Liu R, Ge Y, et al. Discrete content-aware matrix factorization[C]//Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM,2017:325-334
[41] WANG Haoyu, SHAO Nan, LIAN Defu. Adversarial binary collaborative filtering for implicit feedback[C]// The 33nd AAAI Conference on Artificial Intelligence (AAAI 2019), Accepted, Honolulu, Hawaii, January 2019
[42] Maddison C J, Mnih A, The Y W. The concrete distribution: a continuous relaxation of discrete random variables [J]. arXiv Preprint arXiv:1611.00712,2016
[43] Jang E, Gu S, Poole B. Categorical reparameterization with gumbel-softmax [J]. arXiv Preprint arXiv: 1611.01144,2016
[44] Bengio Y, Léonard N, Courville A. Estimating or propagating gradients through stochastic neurons for conditional computation[J]. arXiv Preprint arXiv:1308.3432,2013
[45] Rolfe J T. Discrete variationalautoencoders[J]. arXiv Preprint arXiv:1609.02200,2016
[46] Shan Y, Zhu J, Mao J C. Recurrent binary embedding for gpu-enabled exhaustive retrieal from billion-scale semantic vectors[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM,2018:2170-2179

[47] Jégou H, Douze M, Schmid C. Product quantization for nearest neighbor search [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011, 33(1): 117-128
[48] Ge T Z, He K M, Ke Q F, et al. Optimized product quantization[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2014, 36(4): 744-755
[49] Norouzi M, Fleet D J. Cartesian K-means [C] // 2013 IEEE Conference on Computer Vision and Pattern Recognition, 23-28 June 2013, Portland, OR, USA, 2013: 3017-3024
[50] Chen Y J, Guan T, Wang C. Approximate nearest neighbor search by residual vector quantization[J]. Sensors, 2010, 10(12): 11259-11273
[51] Ge T, He K, Ke Q, et al. Optimized product quantization for approximate nearest neighbor search [C] // 2013 IEEE Conference on Computer Vision and Pattern Recognition(C) VPR. IEEE Computer Society, 2013
[52] Babenko A, Lempitsky V. Additive quantization for extreme vector compression[C] //2014 IEEE Conference on Computer Vision and Pattern Recognition, 23-28 June 2014, Columbus, OH, USA, 2014: 931-938
[53] Babenko A, Lempitsky V. Tree quantization for large-scale similarity search and classification [C] // 2015 IEEE Conference on Computer Vision and Pattern Recognition (C) VPR, 7-12 June 2015, Boston, MA, USA, 2015: 4240-4248
[54] Martinez J, Clement J, Hoos H H, et al. Revisiting additive quantization [M]. Springer, Cham : European Conference on Computer Vision, 2016: 137-153
[55] Li J, Lan X G, Wang J, et al. Fast additive quantization for vector compression in nearest neighbor search[J]. Multimedia Tools and Applications, 2017, 76 (22): 23273-23289
[56] Guo R, Kumar S, Choromanski K, et al. Quantization based fast inner product search[J]. In Artificial Intelligence and Statistics, 2016: 482-490

Research progress on item recalling in recommender systems

LIAN Defu[1,2] XIE Xing[3] CHEN Enhong[1,2]
1 School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026
2 School of Data Science, University of Science and Technology of China, Hefei 230026
3 Microsoft Research Asia, Beijing 100080

Abstract The rapid development of information technology has led to information overload. Recommendation is one of 他 the most effective ways to solve the information overload. In recent years, the rapid development of deep learning has also led to the advancement of recommender systems, and various deep learning based recommendation algorithms have emerged one after another. However, due to the large number of candidate items and the dynamic evolving of user interests, deep learning based recommendation algorithms suffer from computational burden of online recommendation. Therefore, it is almost impossible for these algorithms to be deployed alone in practice. With the development of deep learning based recommendation, the item recallingtechniques (also called approximated search techniques) has also made significant progress. This paper first introduces the research progress of the item recalling techniques based on the nearest neighbor search, and then discusses the research progress of the item recallingtechniquesbased on the maximum inner product search from the perspectives of indexing, locality sensitive hash, learning to hash and vector quantization.

Key words maximum inner product search; item recalling; nearest neighbor search; recommender systems; collaborative filtering; deep recommendation; collaborative filtering