融合 node2vec 和深度神经网络的隐式反馈推荐模型

何瑾琳 刘学军 徐新艳 毛宇佳

(南京工业大学计算机科学与技术学院 南京 211816)

摘 要 利用隐式反馈信息实现个性化推荐是实用且具有挑战性的研究课题。对如何有效结合辅助信息来解决数据稀疏问题从而实现高效推荐的问题进行了研究,提出了一种融合 node2vec 和深度神经网络的隐式反馈推荐模型。该模型采用一种嵌入元数据的深度神经网络框架(Deep Neural Network Framework with Embedded Meta-data,Meta-DNN),首先将用户和项目的 one-hot 向量进行低维映射,再嵌入元数据信息,并结合 node2vec 的二阶随机游走方法学习网络中的邻居节点,使得相邻节点具有相似的节点表示,同时通过增强相邻用户和项目的平滑度来缓解数据稀疏性;最后使用深度神经网络进一步学习用户对项目的偏好,进而为用户产生推荐。其中,还引入了流行度参数对未知项目进行非平均抽样,优化隐式反馈负采样策略。在 Gowalla 和 MovieLens-1M 两个数据集上的实验表明,所提方法可以明显提高系统的预测性能和推荐质量。

关键词 node2vec,推荐系统,神经网络,深度学习,隐式反馈,元数据

中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.06.005

Implicit Feedback Recommendation Model Combining Node2vec and Deep Neural Networks

HE Jin-lin LIU Xue-jun XU Xin-yan MAO Yu-jia

(School of Computer Science and Technology,Nanjing Tech University,Nanjing 211816,China)

Abstract It is extremely practical and challenging to implement personalized recommendation based on large-scale implicit feedback information. In order to solve the problem of data sparseness and then achieve effective recommendation combining various side information,this paper proposed an implicit feedback recommendation model combining node2vec and deep neural networks. This model utilizes a deep neural network framework with embedded meta-data (Meta-DNN),and maps the users and items vectors in a low-dimensional manner. Wherein,the second-order random walk of node2vec is used to learn neighbor nodes in the network with embedded meta-data so that adjacent nodes have similar node representations. Besides,data sparsity is alleviated via improving the smoothness among neighboring users and items. Finally,deep neural network is used for further learning user preferences for items,thus providing recommendation for users. In addition,the popularity parameter is introduced to perform non-average sampling of unknown items and an implicit feedback negative sampling strategy is optimized. Experimental results on the typical Gowalla and MovieLens-1M data sets demonstrate the prediction performance and recommendation quality of the proposed model compared with state-of-the-art recommendation algorithms.

Keywords Node2vec,Recommender system,Neural networks,Deep learning,Implicit feedback,Meta-data

1 引言

在信息爆炸时代,推荐系统在减轻信息过载方面发挥了巨大作用。个性化推荐系统被广泛应用于电子商务、社交媒体等在线服务网络中。传统推荐的关键是基于过去的交互对用户的项目偏好进行建模。最近对推荐的研究已经从明确的评价转向隐式反馈,如购买、点击、观看等。隐式反馈具有以下特点:不能直接表现出用户的喜好倾向,但收集成本更低,应用场景更广,数据规模更大。在实际生活中,通过隐式反馈为用户探索新的兴趣方向,对用户和企业都具有极大的好处。

目前,推荐系统最常用的方法就是协同过滤,其中矩阵分解(MF)最受欢迎,它将用户和项目映射到共享的潜在空间

到稿日期:2018-06-15 返修日期:2018-09-07 本文受江苏省重点研发计划项目(BE2017617,BE2015697),国家重点研发计划子课题(2017YFC0805605)资助。

何瑾琳(1995-),女,硕士生,主要研究方向为数据挖掘、推荐系统,E-mail:hejinlin00@outlook.com;刘学军(1970-),男,博士,教授,CCF 高级会员,主要研究方向为数据库、数据挖掘、推荐系统等,E-mail:lxj_njgd@163.com(通信作者);徐新艳(1980-),女,讲师,主要研究方向为数据挖掘、智能信息处理;毛宇佳(1994-),男,硕士生,主要研究方向为数据挖掘、推荐系统。

中。针对隐式反馈的协同过滤推荐在现阶段主要面临以下两方面挑战。

(1)丰富的元数据信息

基于个性化的推荐系统具有丰富的元数据信息(属性信息等),它们都影响着推荐结果的准确性。用户的偏好受以下几方面因素的影响:1)用户的自身属性,如年龄、性别等;2)用户的社会关系,特别是在 Facebook、豆瓣这样的社交网络中,用户可以看到朋友的状态;3)随着时间的推移而变化,并且可能遵循某些特定顺序,如中午到餐厅,晚上可能到超市或者电影院等。考虑不同的元数据信息会得到不同的上下文预测。

(2)数据稀疏性

基于个性化推荐的数据稀疏问题一直存在。例如,Netflix 电影推荐中的数据密度只有 1.2%,Foursquare 和 Yelp 项目推荐使用的数据密度甚至在 0.6%左右[1]。此外,推荐系统中用户在大多数情况下只提供隐式反馈[2]。数据的稀疏性等级,直接导致传统协同过滤方法即矩阵分解及其各种扩展方法[3]的局限性。

针对以上问题,本文提出了一种基于 node2vec 和深度神经网络的隐式反馈推荐方法,它通过加入元数据信息,丰富与推荐相关的用户和项目的上下文信息,可以有效应对数据稀疏性带来的挑战。本文主要贡献如下:

(1)提出了一种嵌入元数据的深度神经网络框架(Meta-DNN),其能利用隐式反馈预测用户对项目的偏好;

(2)提出将元数据和基于 node2vec 随机游走的方法进行结合来分别学习用户和项目的上下文特征,以缓解数据稀疏性;

(3)使用深度神经网络为非线性用户和项目建立潜在模型,用于预测用户对项目的偏好,并引入了项目流行度,来改进对未交互项目进行采样的方法。

2 相关工作

目前,使用深度神经网络进行推荐是一个趋势。深度神经网络在图像识别、语音识别、文本处理、用户跨域行为等诸多领域已经被验证是有效的。已有很多工作把深度学习用到了推荐中,如自动编码器已被成功应用在基于评分项目的推荐系统中,用来学习用户表示[4]。这些算法主要关注显式反馈,并仅对评分数据进行建模;同时,在涉及到用户和项目特征之间的交互时,仍采用矩阵分解方式。He 等[5]提出用神经结构代替矩阵分解,以捕捉用户交互数据的复杂结构;但没有考虑加入用户和项目丰富的辅助信息。

基于 word2vec 模型[6]的嵌入式学习算法已经被熟练应用于自然语言学习中。给定一个实例及其上下文,目标是使用映射实例来最小化上下文预测的对数损失。假设$\{(i,c)\}$表示为当前词 i 和对应的上下文 c,C 表示语料库,e_i 是当前词语的特征映射,c'是除了上下文之外的词,w 是模型的参数集合,则其损失函数 f 为:

$$-\sum_{i,c}\log p(c\mid i)=-\sum_{i,c}\left(w_c^{\mathrm{T}}e_i-\log\sum_{c'\in C}\exp(w_c^{\mathrm{T}}e_i)\right) \tag{1}$$

受 skip-gram 的启发,Vasile 等[7]使用元数据嵌入,将项目映射到潜在空间,并计算其间的相似性。Handcock 等[8]提出了一种聚类方法来学习潜在社交网络中的社交状态,从而预测社交关系。Tang 等[9]把语言模型和无监督学习从单词序列扩展到图上,提出把网络结构当作文档的集合,将节点当作单词,从网络中通过随机游走产生“句子”,通过 skip-gram 对节点进行向量化。该方法现多被用于链路预测和节点分类问题。Perozzi 等[10]提出 deepwalk 方法,将 word2vec 思想用于图节点一阶相似性的表示学习中。

不同的邻居节点采样方法会生成大不相同的上下文结构。Grover 等[11]在此基础上提出 node2vec 方法,该方法能在一定程度上调节随机游走的走向。他们认为邻居节点具有相似性和同质性,如图 1 所示。节点随机游走学习得到的向量应该能表示两种信息:1)相同网络社区的节点映射到相近的位置,如 u 和 s_1;2)充当类似角色的两个节点的映射方式相似,如 u 和 s_6。这种随机游走方式使两个连接通路较多的节点在向量化后距离很近。

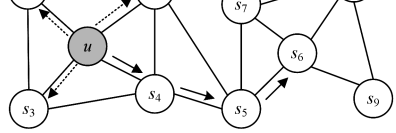


图 1 节点 u 的 BFS 和 DFS 游走策略

Fig. 1 BFS and DFS search strategies of node u

Yang 等[12]提出了基于网络表示学习的半监督学习方法,但其只在一个图上学习节点向量。上述网络表示学习方法均使用单一的网络结构信息,没有充分利用节点的元数据信息。

此外,针对元数据信息,已经衍生出许多基于协同过滤的混合模型[13-14]。目前缺乏一个通用且能够考虑不同上下文信息并从中获得满意推荐的深度神经网络框架。

本文将基于图的网络表示学习方法应用到推荐系统中,尝试学习用户和项目节点的向量表示。利用用户和项目的元数据信息(如属性等),为节点定义新的边权重,以此挖掘出用户和项目更为丰富的上下文信息;利用 skip-gram 模型更新上下文节点的特征向量表示,增强用户-用户之间、项目-项目之间的平滑度,在一定程度上缓解了数据稀疏性。本文研究的推荐工作在端到端的深度神经网络中对用户上下文表示、项目上下文表示以及项目预测进行联合训练,以此优化学习参数,从而得到用户对项目更为精确的偏好预测。

3 嵌入元数据的深度神经网络框架

为了更好地处理数据稀疏性和用户冷启动问题,本文给出一种嵌入元数据的深度神经网络框架。模型训练主要分为 4 步:1)向量低维映射;2)预测用户和项目的上下文信息;3)预测用户对项目的偏好;4)联合训练。

本节将描述各部分的相关计算,并给出整体实现流程,如图 2 所示。

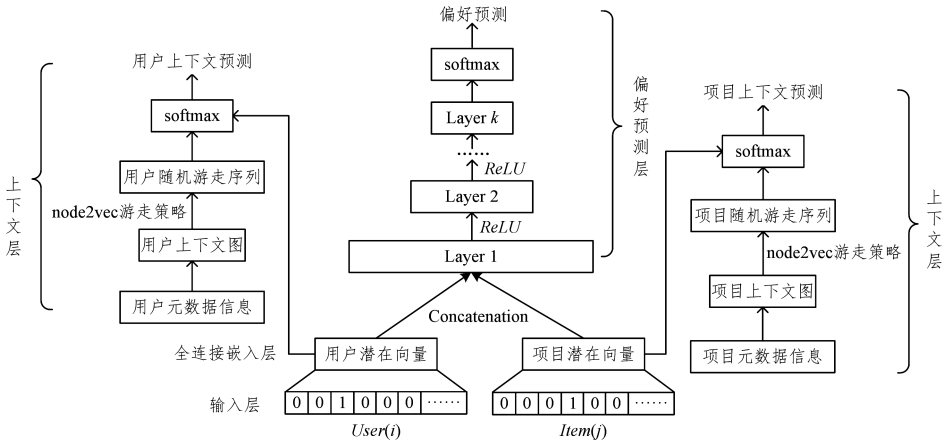


图 2　Meta-DNN 模型

Fig. 2　Meta-DNN model

3.1　向量低维映射

将 one-hot 形式的用户和项目作为输入向量，使用 one-hot 形式的通用特征表示，可以容易地结合其他内容特征来表示用户和项目。输入层上面是一个全连接嵌入层，$\boldsymbol{E}_u \in \mathbb{R}^{N\times D_u}$ 和 $\boldsymbol{E}_v \in \mathbb{R}^{M\times D_v}$ 分别表示用户和项目的潜在特征矩阵，每行分别代表一个用户和一个项目，D_u，$D_v \ll \min(M,N)$。该潜在特征矩阵将输入层的稀疏表示映射为一个稠密向量，获得的用户(项目)嵌入可以看作是用于描述用户(项目)的潜在向量，如 $\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i$ 和 $\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j$ 分别表示用户 u_i 和项目 v_j 的潜在向量。

全连接层的参数表示为 Θ_{embed}，初始化 Θ_{embed} 对用户和项目进行潜在因子建模；然后将用户和项目的潜在向量分别输入到上下文层，通过结合元数据，利用 node2vec 的上下文预测来保留用户和项目之间的上下文信息；再将全连接嵌入层中用户和项目的潜在向量进行融合，并将结果反馈到具有多个隐藏层的前馈神经网络中；在偏好预测层模拟用户和项目之间的交互，通过对偏好预测层顶部的输出层产生的偏好预测进行训练，学习用户对项目的偏好并更新 Θ_{embed}。利用更新后的 Θ_{embed} 可以再对上下文层进行训练，如此循环迭代。

3.2　基于元数据的上下文预测

项目元数据与用户元数据是指描述用户和项目的一些基本信息，它们都有可能影响推荐结果。上下文层的作用是使用 node2vec 方法，从基于元数据的上下文信息学习到用户和项目的向量表示，相邻的节点具有相近的映射，从而缓解数据稀疏性。

3.2.1　上下文图的构造

用户元数据上下文图 $A_U=\{V_U,E_U,W_U\}$ 中，V_U 是用户节点集合；E_U 是用户节点之间的边，它结合了用户朋友关系、个人信息等与用户相关的元数据信息，如具有社交关系即好友的用户之间可以有边，同一个年龄段的用户之间可以有边，具有相同兴趣爱好的用户之间可以有边等；W_U 表示用户边权重。

项目元数据上下文图 $A_V=\{V_V,E_V,W_V\}$ 中，V_V 是项目节点集合；E_V 是项目节点之间的边，它结合了与项目相关的元数据信息，如在电影推荐中，相同类型的电影之间拥有边，同一个演员参与的影视作品之间拥有边；V_U 表示项目边权重。

以用户为例，若其上下文包括 n 种类型的数据信息，分别为 $(R_1,R_2,\cdots,R_n)$，i 和 j 两个节点之间有边 e_{ij}，若 i 和 j 存在 m 种关系，$m \leqslant n$，则 i 和 j 在图中的边权重 $w(i,j)\in W_U$ 为：

$$w(i,j)=\begin{cases}\dfrac{m}{n}, & e_{ij}\in E_U\\ 0, & \text{otherwise}\end{cases} \tag{2}$$

3.2.2　邻居节点的确定

在推荐系统中，热门用户(可能是专家)和热门项目往往是最重要的样本，考虑边权重可以使节点尽量往热门节点方向游走。

在用户元数据上下文图 A_U 上，从 N 个用户中选出长度为 S 的随机游走序列。随机选定源节点 u_1，已知游走路径 $u_{k-2}\rightarrow u_{k-1}$，使用游走概率 $Pro(k|k-1)$ 决定下一步 $u_{k-1}\rightarrow u_k$ 的走向，如图 3 所示。

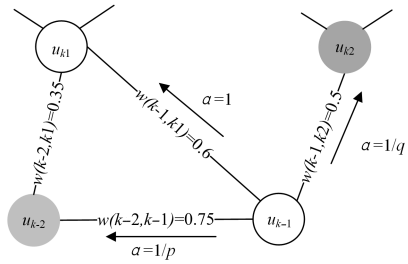


图 3　node2vec 的随机游走过程

Fig. 3　Random walk procedure of node2vec

采用式(2)定义的边权重，以 $N(k-1)$ 表示与节点 u_{k-1} 直接连接的邻居节点集合，得到在随机游走过程中从节点 $k-1$ 直接到节点 k 的游走概率 $Pro(k|k-1)$ 为：

$$Pro(k|k-1)=\alpha * \frac{w(k-1,k)}{\sum\limits_{v\in N(k-1)} w(k-1,v)} \tag{3}$$

其中，α 是二阶随机游走参数。node2vec 随机游走定义了 p 和 q 两个参数变量来调节游走方向，游走到节点 u_{k-1} 时，首先计算其邻居节点与上一节点 u_{k-2} 的距离 d_{tx}，从而得到 α。

$$\alpha=\begin{cases}\dfrac{1}{p}, & d_{k-2,k}=0\\ 1, & d_{k-2,k}=1\\ \dfrac{1}{q}, & d_{k-2,k}=2\end{cases} \tag{4}$$

其中，$d_{k-2,k}=0$ 表示回到 u_{k-2} 本身；$d_{k-2,k}=1$ 表示 u_{k-2} 与 u_k 相连；$d_{k-2,k}=2$ 表示 u_{k-2} 与 u_k 不直接相连，但 u_{k-1} 与 u_k 直接相连。通过调节 p 和 q，可以控制随机游走的方向，两个极端的方向是深度优先遍历(DFS)和广度优先遍历(BFS)，即一直在节点 u 旁边或者一直往外面走。

一个节点获得其邻居序列后，就得到了一个类似句子的向量表示；接下来就是对生成的序列进行训练。

3.2.3 上下文的预测

根据 word2vec[6] 中的推导，给定一个用户及其上下文即邻居节点，基于 skip-gram 模型，式(1)可以表示为如下基于用户元数据上下文图的损失函数。

$$L_{Context_u}=-\sum_{u_i,u_c}\log p(u_c\mid u_i)$$
$$=-\sum_{u_i,u_c}(\boldsymbol{\phi}_c^{\mathrm{T}}\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i-\log\sum_{u_{c'}\in C_u}\exp(\boldsymbol{\phi}_{c'}^{\mathrm{T}}\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i)) \tag{5}$$

同样，可以得到基于项目上下文的损失函数：

$$L_{Context_v}=-\sum_{v_j,v_c}\log p(v_c\mid v_j)$$
$$=-\sum_{v_j,v_c}(\boldsymbol{\phi}_c^{\mathrm{T}}\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j-\log\sum_{v_{c'}\in C_v}\exp(\boldsymbol{\phi}_{c'}^{\mathrm{T}}\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j)) \tag{6}$$

其中，u_c 是 u_i 的上下文；v_c 是 v_j 的上下文；C_u 类似于自然语言中的语料库，是 N 个用户的上下文集合；C_v 则是 M 个项目的上下文集合。

以用户节点为例，$\boldsymbol{\phi}_c^{\mathrm{T}}$ 是节点 u_c 作为节点 u_i 上下文的特征向量表示，$\boldsymbol{\phi}_c^{\mathrm{T}}$ 则是节点 u_i 所有非上下文的特征向量表示。$\boldsymbol{\Phi}=\boldsymbol{\varphi}_c^{\mathrm{T}}\cup\boldsymbol{\phi}_c^{\mathrm{T}}$，表示在 skip-gram 模型中，用户上下文的 N 组参数集合，它是一个特征权重矩阵。本文的目标是通过最小化 $L_{Context_u}$ 来更新 $\boldsymbol{\Phi}$。

Skip-gram 的关键是上下文的预测，即需要最大限度地减少所有用户上下文对的损失，使得邻居重构得到的向量表示尽可能与原本的向量表示接近，以保证共享更多相似上下文的用户与经常一起出现的用户具有更相似的向量表示。对项目数据做类似的处理，参数为 $\boldsymbol{\Psi}=\boldsymbol{\varphi}_c^{\mathrm{T}}\cup\boldsymbol{\phi}_c^{\mathrm{T}}$。

在推荐系统中，用户的数量非常多，若对一个用户的所有非上下文用户都进行参数更新，会给计算造成很大困难。因此，基于 skip-gram 模型，本文使用 Mikolov 等[6]提出的负采样方法进行优化加速，提高效率。

同样以用户为例，从邻居节点样本中抽取 (u_i,u_c)，损失函数可以表示为：

$$L_{Context_u}=-(\log\sigma(\boldsymbol{\phi}_c^{\mathrm{T}}\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i)+$$
$$\sum_{t=1}^{T}E_{c'\sim P_u(c')}\log(-\sigma(\boldsymbol{\phi}_{c'}^{\mathrm{T}}\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i))) \tag{7}$$

其中，$\sigma(x)$ 是 sigmoid 函数，$\sigma(x)=1/(1+e^{-x})$；T 表示负采样次数；$E_{c'\sim P_u(c')}$ 表示负采样实例 c' 服从分布 $P_u(c')$。假设 $d_{c'}$ 表示 c' 在所有随机游走序列中出现的次数，则：

$$P_u(c')\propto d_{c'}^{0.75} \tag{8}$$

假设每一个用户节点随机游走生成长度为 S 的序列，设置滑动窗口大小为 d，从 $\{(S_i,S_k):|i-k|<d\}$ 中抽取一个正对 (u_i,u_c)，得到 $u_i=S_i$，$u_c=S_k$；再根据式(8)的采样方法从所有非滑动窗口用户中抽取 T 个负对。式(7)的损失函数可以简化表示为：

$$L_{Context_u}=-\sum_{(u_i,u_c)}\log\sigma(\boldsymbol{\phi}_{c\cup c'}^{\mathrm{T}}\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i) \tag{9}$$

3.3 用户对项目的偏好预测

用户对项目的偏好预测过程是将包含了上下文关系的用户和项目，基于过去的交互，使用深度神经网络来对用户偏好进行建模，从而预测用户对产品的偏好。

假设用户和项目映射到相同的潜在空间。输入特征向量 $\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i$ 和 $\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j$，可以使用矩阵分解计算两个向量的简单元素之间的内积：

$$h(\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i,\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j)=\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i\odot\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j \tag{10}$$

然而，简单地将潜在特征的乘积进行线性组合的内积不足以捕捉交互数据之间的复杂结构；并且在学习节点表示时，用户和项目被置于不同的上下文图中。本文采用 He 等[5]提出的方法，在生成最终预测之前插入多个非线性的隐藏层。

常见的网络结构设计方法是采用塔式模型[15]，即在更高层使用更少的隐藏单元，这样可以从数据中学习到更多的抽象特征。本文设定：相比前一层，更高的层缩减一半的规模。

Θ_{hidden} 是神经网络的参数，它将潜在特征向量映射为预测分数并在训练过程中不断更新。Θ_{embed} 是全连接层的参数，它将 one-hot 向量映射为潜在特征向量并在训练过程中不断更新。综上，得到用户 u_i 对项目 v_j 的偏好预测为：

$$\hat{y}_{ij}=h(\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i,\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j\mid\Theta_{\text{embed}},\Theta_{\text{hidden}}) \tag{11}$$

交互函数 h 是一个多层神经网络，可以被表示为：

$$h(\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i,\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{E}_j)=h^{\text{out}}(h^F\cdots(h^1(h^0(\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i,\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j)))\cdots) \tag{12}$$

其中，h^{out} 表示输出层的激活函数，F 表示隐藏层的总数。

将用户和项目的特征向量通过矢量连接进行合并，得到 $\begin{bmatrix}\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i\\ \boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j\end{bmatrix}$，将其反馈到具有多个隐藏层的前馈神经网络中。因此，第一个隐藏层的输入为：

$$h^0(\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i,\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j)=\begin{bmatrix}\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i\\ \boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j\end{bmatrix} \tag{13}$$

假设 W^f 和 b^f 是分别第 f 层的权重矩阵和偏置向量，则第 f 层的输出可以表示为：

$$h^f(\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i,\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j)=ReLU(W^fh^{f-1}(\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i,\boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j)+b^f) \tag{14}$$

采用 $ReLU$ 作为激活函数，它支持稀疏激活，更适合稀疏的数据，使模型不至于过拟合。

$$ReLU(\boldsymbol{x})=\max(0,\boldsymbol{x}) \tag{15}$$

其中，$\boldsymbol{x}$ 表示输入向量。

为了特别处理推荐中隐式反馈的性质，使用概率函数 sigmoid 作为激活函数作用在输出层 h^{out}。预测分数 $\hat{y}_{ij}$ 表示 v_j 与 u_i 相关的可能性大小，由此将输出限制到[0,1]的范围。结合式(12)－式(15)，可以得到如下公式：

$$\hat{y}_{ij}=\sigma(ReLU(W^F(\cdots ReLU(W^1\begin{bmatrix}\boldsymbol{E}_u^{\mathrm{T}}\boldsymbol{u}_i\\ \boldsymbol{E}_v^{\mathrm{T}}\boldsymbol{v}_j\end{bmatrix}+b^1)\cdots)+b^F)) \tag{16}$$

学习模型参数时，现有的误差损失函数主要运用均方误差(squared loss)进行回归：

$$L_{\text{Preference}}=\sum_{u_iv_j\in I\cup I^-}w_{ij}(\hat{y}_{ij}-y_{ij})^2 \tag{17}$$

其中，w_{ij} 表示训练实例 (u_i,v_j) 的权重。但均方误差多用于显式反馈，不适合处理二值的隐式反馈。

为了充分利用隐式反馈信息，本文将最终预测的值看作二分类问题，y_{ij} 的值是二进制值，u_i 与 v_j 有交互时 y_{ij} 为 1，否则 y_{ij} 为 0。采用逻辑回归进行训练，得到如下对数损失函数：

$$\begin{aligned} L_{\text{Preference}} &= \log p(I, I^- \mid \Theta_{\text{embed}}, \Theta_{\text{hidden}}) \\ &= -\sum_{u_i, v_j \in I} \log \hat{y}_{ij} - \sum_{u_i, v_j \in I^-} \log(1-\hat{y}_{ij}) \\ &= -\sum_{u_i, v_j \in I \cup I^-} y_{ij} \log \hat{y}_{ij} + (1-y_{ij}) \log(1-\hat{y}_{ij}) \end{aligned} \tag{18}$$

其中，I 表示该用户与这个项目有交互；未交互的项目 I^- 包括了未知数据和真正不相关的数据，并不能表示该用户对这个项目不感兴趣。若对 I^- 采用均匀采样的方法，则会引入较大的数据噪声。

本文根据项目流行度对未交互数据采用了非均匀采样策略。项目流行度 f_i 表示为：

$$f_i = \frac{|R_i|}{\sum_{j=1}^{M} |R_j|} \tag{19}$$

其中，$|R_i|$ 表示项目 i 与所有用户交互的总次数，M 为项目总数。一个项目的流行度越高，其被用户选择交互的概率越大，因此有理由认为错过流行项目更可能是因为用户对该项目不感兴趣（而非未知）。根据项目流行度，设定项目 i 是真正不相关数据的概率为：

$$c_i = \frac{f_i^{\varepsilon}}{\sum_{j=1}^{M} f_j^{\varepsilon}} \tag{20}$$

其中，ε 表示流行项目与非流行项目的权重分布。$\varepsilon > 1$，表示流行项目的权重被提升，应加大其与非流行项目的区别；$\varepsilon \in (0,1)$可以抑制流行项目的权重。根据 c_i 的大小，对 I^- 中的数据进行负采样。

3.4　联合训练

使用 TensorFlow[1] 来联合训练 Meta-DNN 模型，并采用随机梯度下降方法（SGD）进行优化。

Meta-DNN 输出了 3 组预测，即用户对项目偏好预测、用户上下文预测和项目上下文预测，通过优化 3 个损失函数和来共同训练模型：

$$L_{\text{mix}} = \omega(L_{Context_u} + L_{Context_v}) + (1-\omega) L_{\text{Preference}} \tag{21}$$

其中，$L_{Context_u}$ 和 $L_{Context_v}$ 是增强用户和项目上下文平滑度的正则化损失项，目标是通过正则化基于元数据上下文图的潜在因子模型来增强相似用户之间和相似项目之间的平滑性；$L_{\text{Preference}}$ 是用户对项目的偏好预测损失项；ω 实现三者之间的平衡。

Meta-DNN 的具体步骤如算法 1 所示。

算法 1　Meta-DNN 迭代训练算法

输入：用户元数据上下文图 $A_U = \{V_U, E_U, W_U\}$；项目元数据上下文图 $A_V = \{V_V, E_V, W_V\}$；用户数据 U、项目数据 V、参数 ω；batch size B；迭代训练次数 Iteration；随机游走长度 S；滑动窗口大小 d；初始化 $\Phi, \Psi, \Theta_{\text{embed}}, \Theta_{\text{hidden}}$

输出：$\Phi, \Psi, \Theta_{\text{embed}}, \Theta_{\text{hidden}}$。

Initialize walks_u，walks_u to Empty

For it from 1 to Iteration do

　For all user node$\in V_U$，all item node$\in V_V$ do

　　walk_u = node2vecWalk(A_U，start node u，S)

　　Append walk_u to walks_u

　　walk_v = node2vecWalk(A_V，start node v，S)

　　Append walk_v to walks_v

　end for

　$L_{\text{Context}_u} = -\frac{-\omega}{B} \sum_{u_i, u_c} \log \sigma(\boldsymbol{\phi}_{c \cup c'}^{\mathrm{T}} \mathbf{E}_u^{\mathrm{T}} \mathbf{u}_i)$

　Update Φ by SGD

　$L_{\text{Context}_v} = -\frac{-\omega}{B} \sum_{v_j, v_c} \log \sigma(\boldsymbol{\varphi}_{c \cup c'}^{\mathrm{T}} \mathbf{E}_v^{\mathrm{T}} \mathbf{v}_j)$

　Update Ψ by SGD

　$\hat{y}_{ij} = h(\mathbf{E}_u^{\mathrm{T}} \mathbf{u}_i, \mathbf{E}_v^{\mathrm{T}} \mathbf{v}_j \mid \Theta_{\text{embed}}, \Theta_{\text{hidden}})$

$L_{\text{Preference}} = -\frac{1-\omega}{B} \sum_{u_i v_j \in \mathcal{Y} \cup \mathcal{Y}^-} y_{ij} \log \hat{y}_{ij} + (1-y_{ij}) \log(1-\hat{y}_{ij})$

　Update $\Theta_{\text{embed}}, \Theta_{\text{hidden}}$ by SGD

end for

4　实验

本文采用的实验环境为：32 GB 内存，Intel(R) Xeon(R) E5-2630v3 处理器，2.40GHz；GPU 为 NVIDIA GeForce GTX 1080 Ti；操作系统为 Windows 10；实现语言为 Python。

在实验设计和结果分析部分，基于 TensorFlow 来实现本文提出的方法。分别在 CPU 和 GPU 上实现它的高效运行，其中 GPU 比 CPU 上的运行速度快 10 倍以上。当批大小 B 增加到几百时，可以更快地计算并达到收敛，尤其是在 GPU 上。在默认参数设置下，GPU 完成每次迭代的时间大约为 113 s。

下面首先介绍数据集，并说明相应的评价标准及对比算法；再给出所提模型与其他方法的对比实验结果，以及所提模型在不同参数设置下的对比实验结果，并对实验结果进行相应的分析。

4.1　数据集

本文使用两个可公开访问的数据集：Gowalla[2] 和 MovieLens[3]（ML-1m）。数据集的基本特征数据如表 1 所列。

表 1　数据集的基本信息

Table 1　Basic information of data sets

数据集	用户数	项目数	交互数	稀疏度/%
Gowalla	30887	18995	860888	99.87
ML-1m	6040	3952	1000209	95.53

Gowalla：这个隐式反馈数据集被广泛用于用户兴趣点的推荐中。过滤掉少于 15 次签到的用户，同时要求每个兴趣点至少被用户访问 10 次。包括的用户元数据为好友关系和签到时间；项目元数据为兴趣点类别和兴趣点的地理位置等。

ML-1m：该电影评级数据集被广泛用于评估协同过滤算法。每个用户至少有 20 个评分。这是显性反馈数据集，需要

1) https://www.tensorflow.org/

2) 引用参考文献[16]

3) https://grouplens.org/datasets/movielens/

将其转换为隐式数据。其中，被标记为 0 表示用户未对该项目进行评分，标记为 1 表示用户已对该项目进行评分。包含的用户元数据有年龄、职业、评分时间；项目元数据有电影类别。

对每个用户，按照8:2的比例将所有交互数据随机分成训练集和测试集，重复 5 次。

4.2 评价指标

采用平均正确率均值（Mean Average Precision，MAP）[17]来评测 TOP-K 项目的推荐性能，MAP 越大，表示算法的推荐准确率越高。MAP 是对所有测试用户的平均正确率（Average Precision，AP）的再一次平均。给定一个用户 i 及长度为 K 的已排序推荐列表 $\langle j_1, j_2, \cdots, j_K \rangle$，$pref(i) \in \{0,1\}$，测试集在算法排序后的第 i 项为正例项时 $pref(i)=1$，否则 $pref(i)=0$。假设用户 i 选择了其中 N 个，则可以计算出平均正确率：

$$AP@K=\frac{\sum_{k=1}^{K} Pre@k \times ref(k)}{N}, Pre@K=\frac{\sum_{t=1}^{K} pref(t)}{K}$$

采用折扣累计利润及其归一化（normalized Discounted Cumulative Gain，nDCG）来衡量用户对推荐列表的满意程度。如果相关的项目排到前面，$nDCG$ 会较大。假设 K^+ 表示在该用户所有有交互的项目里取前 K 个。

$$nDCG@K=\frac{DCG@K}{\sum_{i=1}^{K^+} \frac{1}{\log_2(i+1)}}, DCG@K=\sum_{i=1}^{K} \frac{pref(i)}{\log_2(i+1)}$$

由于在评估过程中为每个用户排列所有项目花费的时间太长，因此在探究 Meta-DNN 在不同的要素和参数设置下的性能时，采用 leave-one-out 方法进行评估[5]。对每一个用户，抽取一个 ground-truth（GT）作为测试集，随机抽取 100 个与用户没有交互的项目，与测试集组成 101 个候选列表。排序列表的性能使用 HR@ 10 来评估，该指标被广泛应用在 TOP-K 推荐中，表示 GT 是否出现在了 TOP-10 列表中。

$$HR@K=\frac{\text{number of } Hits@K}{|GT|}$$

4.3 模型对比

本文选定了 4 种方法作为对比模型。

（1）eALS[2]：使用平方损失的最优 MF 方法。优化了式（19），将所有未交互的数据视为消极实例，改进了采样策略，适用于隐式推荐。

（2）NeuMF[5]：将多层感知机与协同过滤相结合，也是与本文最相关的工作，但它未考虑使用元数据和节点向量表示方法来学习上下文信息。

（3）BPR[18]：优化了使用式（11）的 MF 模型，将推荐问题转化为成对排序问题。从未观测到的数据中为每个积极实例随机抽样，得到一个消极实例形成对，其可用于隐式反馈推荐。

（4）iMF[19]：基于隐式反馈的矩阵分解。将 0-1 矩阵分解成功应用于隐式反馈推荐。

本文同时设计实验来验证利用基于元数据的用户上下文和项目上下文所带来的好处，比较以下 4 种模型。

（1）DNN-P：使用神经网络预测用户对项目的偏好。

（2）Meta-DNN-PU：融合基于用户元数据的上下文进行偏好学习。

（3）Meta-DNN-PI：融合基于项目元数据的上下文进行偏好学习。

（4）Meta-DNN-PUI：基于用户和项目元数据的上下文进行用户对项目的偏好学习，即本文提出的方法。

出于实验效果和有效性的考虑，将参数设置为以下默认值：对初始化训练的 Meta-DNN 模型，采用高斯分布随机初始化模型参数，平均值为 0，标准差为 0.01。深度神经网络中的隐藏层数设置为 3，每一层大小分别设置为 32，16，8；流行项目权重 $\varepsilon=0.5$，无交互的项目的采样数为 5。在邻居节点采样过程中，随机游走长度 $S=10$，滑动窗口 $d=3$，负采样数 $T=5$，$p=q=1$。潜在因子矩阵维度设为 16。联合训练过程中，损失函数权重 $\omega=0.25$，批大小设为 256，学习速率为 0.0001。

除此之外，4.4 节还评估了不同参数对 Meta-DNN 性能的影响。

4.4 实验结果及性能分析

本文从 3 个角度来评估 Meta-DNN 模型。

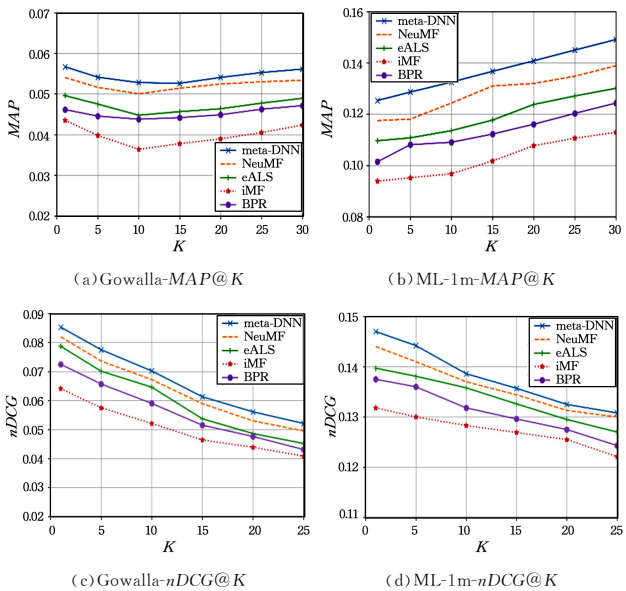
（1）将 Meta-DNN 模型与 4 种现有的隐式反馈推荐模型进行比较。

（2）分析基于元数据的用户上下文、基于元数据的项目上下文对推荐结果的贡献。

（3）讨论相关参数对推荐结果的影响。

4.4.1 推荐模型的比较与分析

图 4 显示了各推荐算法在不同数据集上的 TOP-K 推荐性能。由于 Meta-DNN 模型融合了用户元数据、项目元数据以及偏好预测，因此其相比于其他 4 个模型可以显著提高 Top-K 推荐的命中率。



（a）Gowalla-$MAP@K$ （b）ML-1m-$MAP@K$

（c）Gowalla-$nDCG@K$ （d）ML-1m-$nDCG@K$

图 4 各算法在两个数据集上的性能对比

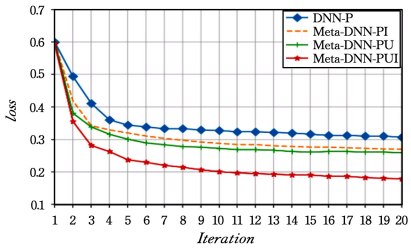
Fig. 4 Performance comparison of algorithms on two datasets

Meta-DNN 在两个数据集上都表现出了最好的性能，超过了最先进的推荐方法 NeuMF（在 Gowalla 上，两个指标均有约 4.9%的相对提升；在 ML-1m 上，两个指标均有约 6.3%

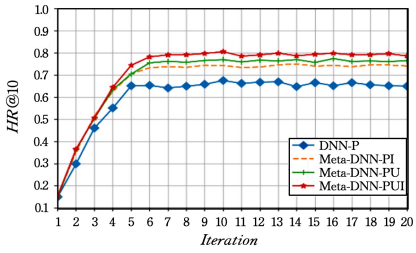
的相对提升）。可以看出，Meta-DNN 方法整体上表现出了明显的优越性，说明该方法可以显著提高 TOP-K 推荐的命中率和准确率。两种表现最好的方法 NeuMF 和 Meta-DNN 都使用了深度神经网络结构，表明了神经嵌入可以通过隐式反馈数据准确地对用户和项目之间的交互建模，而结合大量元数据的上下文信息可以提高推荐的精度。iMF 的性能表现最差，这是因为 iMF 专注于优化误差平方和，过分关注整体误差而忽视了排在前几位的命中率。

4.4.2　要素影响分析

图 5 显示了 Meta-DNN 方法在 MovieLens 上每次迭代的训练损失和 HR@10 的推荐性能。当迭代次数增加时，训练损失逐渐减少，预测性能提高。前 10 次迭代的影响尤为显著，过多的迭代可能使模型过拟合（图中只画出了前 20 次）。其次，在 4 种对比方法中，Meta-DNN-PUI 达到了最低的训练损失，其次是 Meta-DNN-PU 和 Meta-DNN-PI，然后是 DNN-P。推荐性能显示出 Meta-DNN-PUI＞Meta-DNN-PU＞Meta-DNN-PI＞DNN-P 的趋势。由上述实验结果也可以看出，学习用户和项目的上下文信息对项目推荐是至关重要的，它们的融合有助于提高推荐精度，增强模型的鲁棒性。



（a）损失函数训练



（b）HR@10

图 5　迭代次数对训练损失和 HR@10 的影响

Fig. 5　Impact of iteration number on training loss and HR@10

4.4.3　参数分析

Meta-DNN 模型有以下几个重要的参数。

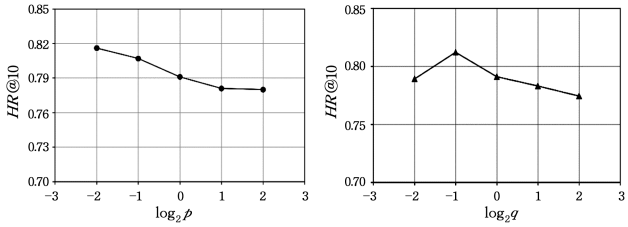
（1）上下文图中节点的随机游走参数 p 和 q；

（2）隐藏层的数量 F 以及容量 h；

（3）流行项目权重 ε 以及负采样的个数。

研究这些参数时，其他参数采用默认值，从而分析其对 Meta-DNN 性能的影响。

设置 $p,q\in\{0.25,0.5,1,2,4\}$ 进行网格搜索，$p=q=1$ 时为无偏向的随机游走即 Deepwalk。图 6 给出基于 MovieLens 数据集在 HR@10 这个评估指标上对不同参数 p 和 q 的对比结果。Gowalla 上也显示出了类似的趋势。可以看出，Meta-DNN 的性能受到参数 p 和 q 的影响。在 MovieLens 上，最好的探索策略为 $p=0.25$，$q=0.5$，低 q 值表示支持节点向外探索与之结构相似的节点，但低 p 值可以保证获取的邻居节点不会离开始节点太远。该实验验证了 3.2 节提出的同质性和结构性对节点的影响。不同的 p，q 值可以影响推荐精度。



（a）参数 p　　（b）参数 q

图 6　随机游走参数 p，q 与 HR@10 的关系

Fig. 6　Relationship between random walk parameters p and q with HR@10

本文通过改变隐藏层的数量及容量来验证使用深度神经网络进行偏好学习的有效性。在 MovieLens 上的实验结果如表 2 所列。Gowalla 上显示出了类似的趋势。

因为深度神经网络中的最后一层隐藏层决定了模型的性能，所以将其作为预测因子 h，并分别使用[4，8，16，32]的大小进行评估。若模型容量是 32 且隐藏层数 $F=3$，则深度神经网络的结构为 32→16→8。用户、项目潜在因子维度为 16。

表 2　不同隐藏层数的神经架构模型在 HR@10 上的性能对比

Table 2　Comprehensive performance of HR@10 with neural model on different hidden layers

h	$F=0$	$F=1$	$F=2$	$F=3$	$F=4$
4	0.535	0.712	0.741	0.758	0.769
8	0.537	0.735	0.764	0.779	0.788
16	0.536	0.743	0.782	0.791	0.796
32	0.537	0.746	0.789	0.795	0.801

从表 2 中可以发现，对具有相同能力的模型，更多的隐藏层有利于性能的提升，这表明了利用深层模型进行推荐的有效性。层数为 3 时几乎可以获得最佳性能；层数为 4 层时，性能提升并不明显。对于没有隐藏层（即嵌入层直接映射到输出层）的 $F=0$，HR@10 性能非常弱，从而验证了 3.3 节提出的观点，即对用户和项目的潜在向量使用内积连接不足以对其特征的相互作用进行建模，需要使用隐藏层进行变换。

除了隐藏层的数量外，每层的预测因子个数也是模型中的一个敏感参数。顶层预测因子为 32 时，模型可以获得最佳性能。以上表明：对于稀疏数据，包含更多的预测因子可以达到更好的推荐效果。

3.3 节还加入了项目流行度来对未交互的项目进行非均匀负采样，图 7 是参数 ε 以及不同负采样率对推荐性能的影响。从图 7 可以看出，当 ε 取值为 0.4 或 0.5 时，能达到模型的最佳性能；超过这个范围时，性能开始下降，这可能是因为过度加重流行项目作为消极实例的影响。因此，需要对非流行的项目进行适当的权重计算。此外，还加入不同负采样个数来观察实验性能，neg-3 表示对一个已交互的项目选取 3 个未交互的项目作为负样本，结果表明抽取更多的负样本对提高实验性能有帮助。对于 MovieLens 数据集，最佳的负采样个数大致是 5，这与之前的工作结果一致。

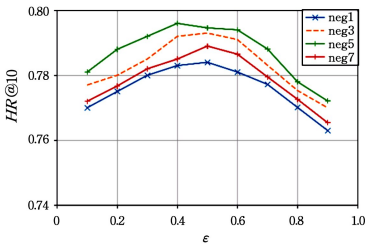


图 7 流行项目权重 ε 和负采样个数对 HR@10 的影响

Fig. 7 Impact of popular items ε and number of negatives on HR@10

结束语 本文对使用隐式反馈进行项目推荐进行了研究,提出了一种基于 node2vec 和深度神经网络的推荐算法。一方面,利用基于元数据的上下文图平滑处理上下文信息,缓解数据稀疏性;另一方面,利用神经网络对用户和项目之间的非线性复杂交互进行建模,在基于 Meta-DNN 的神经网络架构上生成最终的预测结果。实验表明,本文提出的方法在多个评价指标上均有较好的表现。

在今后的研究工作中,可以探索复杂有向异构属性上下文图,使模型包含更丰富的上下文信息,用户与项目之间存在的交叉特征也是很重要的推荐信息。此外,用户对项目的偏好还受到时间和空间的影响,将注意力机制考虑在内也是另一个值得研究的方向。

参 考 文 献

[1] LI X,CONG G,LI X L,et al. Rank-GeoFM: A Ranking based Geographical Factorization Method for Point of Interest Recommendation[C]// International ACM SIGIR Conference on Research and Development in Information Retrieval. Santiago: ACM,2015:433-442.

[2] HE X,ZHANG H,KAN M Y,et al. Fast Matrix Factorization for Online Recommendation with Implicit Feedback[C]// International ACM SIGER Conference on Research & Development in Information Retrieval. Pisa: ACM,2016:549-558.

[3] WANG H,WANG N,YEUNG D Y. Collaborative Deep Learning for Recommender Systems[C]// ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Sydney: ACM,2015:1235-1244.

[4] SHENG L,JAYA K,YUN F. Deep Collaborative Filtering via Marginalized Denoising Auto-encoder[C]// ACM International on Conference on Information and Knowledge Management. Melbourne: ACM,2015:811-820.

[5] HE X,LIAO L,ZHANG H,et al. Neural Collaborative Filtering [C]// International Conference on World Wide Web. Perth: ACM,2017:173-182.

[6] MIKOLOV T,SUTSKEVER I,CHEN K,et al. Distributed Representations of Words and Phrases and their Compositionality [J]. Advances in Neural Information Processing Systems, 2013,26:3111-3119.

[7] VASILE F,SMIRNOVA E,CONNEAU A. Meta-Prod2Vec: Product Embeddings Using Side-Information for Recommendation[C]// ACM Conference on Recommender Systems. Boston: ACM,2016:225-232.

[8] HANDCOCK M S,RAFTERY A E,TANTRUM J M. Model-based clustering for social networks [J]. Journal of the Royal Statistical Society,2007,170(2):301-354.

[9] TANG J,QU M,WANG M,et al. LINE: Large-scale Information Network Embedding[C]// Proceedings of the 24th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee,2015:1067-1077.

[10] PERPZZIE B,Al-RFOU R,SKIENA S. Deepwalk:online learning of social representations[C]// Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM,2014:701-710.

[11] GROVER A,LESKOVEC J. node2vec:Scalable Feature Learning for Networks [C]// Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Francisco: ACM,2016:855-864.

[12] YANG Z,COHEN W W,SALAKHUTDINOV R. Revisiting semi-supervised learning with graph embeddings[C]// International Conference on International Conference on Machine Learning. New York: ICML,2016:40-48.

[13] 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.

[14] YAO W,HE J,HUANG G,et al. A graph-based model for context-aware recommendation using implicit feedback data [J]. World Wide Web-internet & Web Information Systems,2015, 18(5):1351-1371.

[15] HE K,ZHANG X,REN S,et al. Deep Residual Learning for Image Recognition[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2016: 770-778.

[16] ZHANG C,SHOU L,CHEN K,et al. Evaluating geo-social influence in location-based social networks[C]// ACM International Conference on Information and Knowledge Management. Maui: ACM,2012:1442-1451.

[17] YIN J,WANG Z S,LI Q,et al. Personalized recommendation based on large-scale implicit feedback[J]. Journal of Software, 2014,5(9):1953-1966. (in Chinese)
印鉴,王智圣,李琪,等. 基于大规模隐式反馈的个性化推荐[J]. 软件学报,2014,5(9):1953-1966

[18] RENDLE S,FREUDENTHALER C,GANTNER Z,et al. BPR: Bayesian personalized ranking from implicit feedback [C]// Conference on Uncertainty in Artificial Intelligence. Montreal: AUAI Press,2009:452-461.

[19] HU Y,KOREN Y,VOLINSKY C. Collaborative Filtering for Implicit Feedback Datasets[C]// IEEE International Conference on Data Mining. Leipzig: IEEE,2009:263-272.