Computer Science and Application 计算机科学与应用, 2022, 12(1), 54-62
Published Online January 2022 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2022.121007

Hans汉斯

基于二分网络表示学习的开源项目推荐方法

林海铭[1]，田春岐[1]，王　伟[2]

[1]同济大学，计算机科学与技术系，上海
[2]华东师范大学，数据科学与工程学院，上海

收稿日期：2021年12月14日；录用日期：2022年1月10日；发布日期：2022年1月17日

摘　要

随着开源生态的迅猛发展，越来越多的开发者加入到开源社区中，以开放、共享、协同的软件开发模式构建开源项目。许多代码托管平台如GitHub上托管着海量的开源项目，涵盖各类技术领域，开发者可以通过关键字搜索自己感兴趣的主题或项目名称进行浏览探索、加入或者代码重用。然而，开发者往往需要花费大量的时间和精力，才能通过关键词准确地描述自身的兴趣与目标项目的特征，找到符合自身需求的开源项目。针对这一问题，本文提出了一种基于二分网络表示学习的开源项目推荐方法。该方法结合开发者在参与项目贡献时的不同参与方式，构建出基于活跃度模型的开源贡献二分网络，并融合开发者与项目之间的显式关系和隐式关系，学习开发者的兴趣和技术偏好，向开发者进行个性化开源项目推荐。实验结果表明，本文所提的方法在推荐20个候选项目时的正确率超过42.37%，能够有效地为开发者推荐其感兴趣的开源项目。

关键词

开源项目，项目推荐，二分网络，表示学习

Open Source Project Recommendation Method Based on Bipartite Network Representation Learning

Haiming Lin[1], Chunqi Tian[1], Wei Wang[2]

[1]Department of Computer Science and Engineering, Tongji University, Shanghai
[2]School of Data Science & Engineering, East China Normal University, Shanghai

Received: Dec. 14th, 2021; accepted: Jan. 10th, 2022; published: Jan. 17th, 2022

文章引用: 林海铭, 田春岐, 王伟. 基于二分网络表示学习的开源项目推荐方法[J]. 计算机科学与应用, 2022, 12(1): 54-62. DOI: 10.12677/csa.2022.121007

Abstract

With the rapid development of the open source ecosystem, more and more developers have joined the open source community to build open source projects with an open, shared, and collaborative software development model. Many code hosting platforms, such as GitHub, host a large number of open source projects, covering various technical fields. Developers can search for topics or projects they are interested in by keywords to explore, contribute, or reuse code. However, developers often need to spend a lot of time and energy to accurately describe their interests and characteristics of target projects through keywords, and find projects that meet their own needs. To solve this problem, this paper proposes an open source project recommendation method based on bipartite network representation learning. This method combines the different participation methods of developers when participating in project contributions, constructs an open-source contribution bipartite network based on the activity model, and integrates the explicit and implicit relationships between developers and projects to recommend open source projects to developers. The experimental results show that the accuracy of the proposed method can achieve over 42.37% when recommending 20 candidate projects, which means it can effectively recommend closely correlated projects to developers.

Keywords

Open Source Projects, Projects Recommendation, Bipartite Network, Representation Learning

Copyright © 2022 by author(s) and Hans Publishers Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
Open Access

1. 引言

随着互联网的发展，软件开发工具与社交媒体[1]正在深度融合[2] [3]。软件开发活动本质上是开发者群体之间沟通交流、灵感激发的创作过程[4]。传统的软件开发模式下，团队外部的大众群体很难参与到开发活动中。因此，以 GitHub 为代表的社交化编程平台基于分布式开发工作流，为开发者提供了极其便利且低成本的沟通交流方式。开发者可以完整的拷贝自己感兴趣的开源项目，开辟新的分支进行独立开发，通过问题追踪系统与代码评议系统自由地与他人交流，并将自己的贡献提交给上游仓库。这种开放、共享、协同的软件开发模式，吸引了大量的开发者自愿加入社区，并与其他开发者建立合作关系，共同推进项目的开发[5]。

尽管开源社区发展迅速，但由于这些开发者来自于世界各地，性格特征、文化与专业知识背景各不相同[6]，因此要从海量的开源软件项目中找到与开发者自身的兴趣与专业特长相匹配的项目，具有一定的挑战性。当前已有的开源项目搜索引擎通常侧重于对检索内容的文本匹配，但因为项目数量极其庞大，开发者的需求许多时候又是隐性的，难以通过关键词准确地描述自身的兴趣与目标软件项目的特征，因此开发者往往需要花费大量的时间和精力，才能找到符合自身需求的开源项目，以便重用其功能或者为其做出贡献。

针对上述问题，本文提出一种基于二分网络表示学习的个性化开源项目推荐方法，该方法基于开源生态系统的演化过程中沉淀下来的海量的开发者贡献与协作行为数据，构建出基于活跃度模型的开源贡献二分网络，并对网络中的显式关系与隐式关系进行学习，为开发者推荐符合其软件开发活动偏好和兴

趣的开源项目，从而提升大规模群体协同的效率，从而促进开源软件生态系统的良性发展。

2. 相关工作

随着开源软件和开源技术的蓬勃发展，对于开源生态系统的研究也成为近年来的热门研究领域。而开源生态系统中的项目推荐，是促进开源生态快速健康发展的实际需求[7]，因此引起了研究者的广泛兴趣。许多工作从项目及开发者的特征入手，应用传统的推荐算法为开发者推荐相关的项目。如 Guendouz 等[8]根据开发者是否参与过项目，构建开发者－项目评分矩阵，然后基于项目协同过滤方法实现 Top-N 推荐。He 等[9]提出了基于内存的协同过滤方法，结合基于项目和基于开发者的协同过滤方法设计个性化推荐系统。为了解决协同过滤方法带来的稀疏性问题，Sun 等人[10]研究了基于 TF-IDF 与模拟退火算法的项目个性化推荐算法，并利用开发者的反馈数据来改进推荐准确率。Zhang 等人[11]基于项目的 Stars 数据和 README 文件，计算项目与项目之间的相似性得分，设计了 RepoPal 系统用于推荐与给定项目相似的项目。Yang 等人[12]则结合了项目自身流行度、关联项目技术相关度以及大众贡献者之间的社交关联度三个维度的特征信息，基于 LTR 方法构建开发者个性化开源项目推荐模型。

3. 基于二分网络表示学习的个性化推荐方法

本节首先给出基于二分网络表示学习的个性化推荐方法框架，然后介绍如何基于活跃度模型构建开源贡献二分网络，接着详细描述了如何建模网络中的显式关系和隐式关系，最后给出了融合显式关系与隐式关系的个性化推荐算法。

3.1. 推荐算法框架

推荐算法框架如图 1 所示。整体算法框架分为四个模块。第一个模块首先从 GitHub 获取开发者历史行为数据；第二个模块主要是基于活跃度模型，通过计算开发者对项目活跃度贡献值的大小，建立开源贡献二分网络；第三个模块对节点间的显式关系和隐式关系进行建模；第四个模块将显式关系和隐式关系的目标函数结合起来，得到网络节点的嵌入向量，计算开发者和项目的相关性得分，最后根据评分对开发者进行项目推荐。

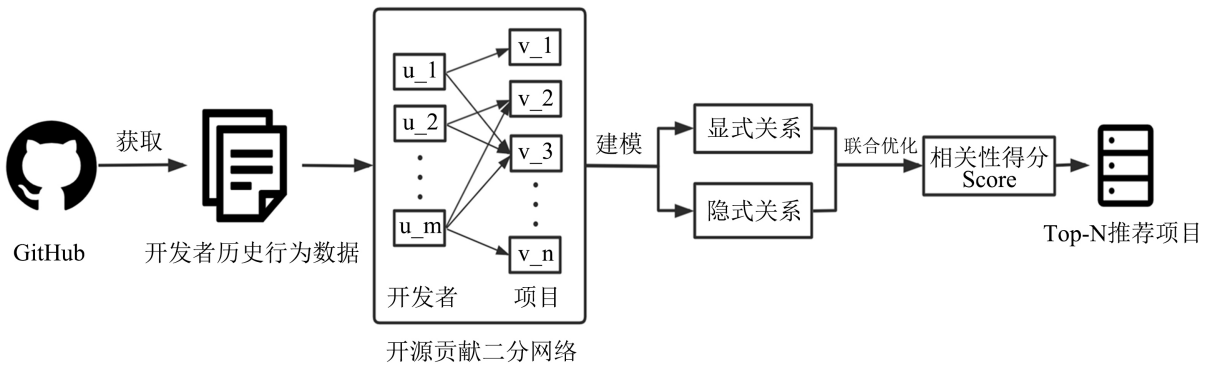

Figure 1. Overview of the open-source project recommendation framework
图 1. 开源项目推荐框架

3.2. 基于活跃度模型的开源贡献二分网络的构建

首先，我们以开发者和项目作为两类节点，构建二分网络。定义开源贡献二分网络 $G=(U,V,E)$，其中 U 和 V 分别表示开发者节点集和项目节点集，$E \subseteq U \times V$ 是网络中边的集合，且 E 仅存在于节点集

U 和 V 之间。每条边带有一个权重值 w_{ij}，表示节点 u_i 和 v_j 之间的连接强度，本文中我们将其定义为开发者对项目的活跃度贡献值大小。图 2 展示了一个开源贡献二分网络的示例。

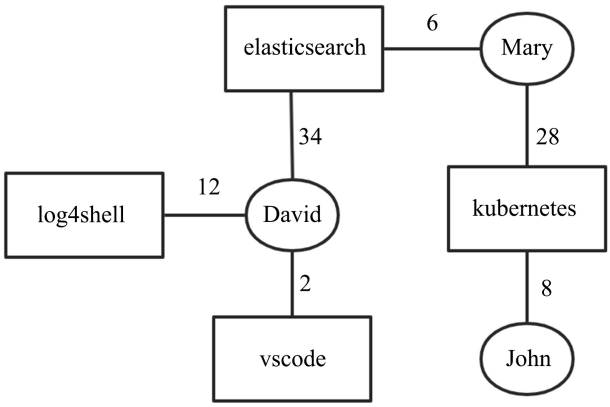

Figure 2. An example of the open-source contribution bipartite network
图 2. 开源贡献二分网络示例

具体地，我们通过统计分析开发者的行为数据来评估某开发者在某具体项目中的活跃情况。根据文献[13]的方法，我们选取了能够反应开发者活跃程度的五项数据指标集合 $C_k=\{c_{ic},c_{oi},c_{opr},c_{prrc},c_{prm}\}$，分别表示 Issue 评论、新建 Issue、新建 PR、PR Review 评论和 PR 合入五种行为事件由该开发者触发的发生次数，具体描述见表 1，这五项数据指标表示开发者在参与项目贡献时的不同参与方式，通常被认为是描述开发者和开源项目活跃程度的重要衡量指标[14]。

Table 1. Metrics for open source repository activity evaluation
表 1. 开源项目活跃度评测数据指标

指标	含义	描述
c_{ic}	Issue Comment	参与到项目某一具体 Issue 下的讨论
c_{oi}	Open Issue	在项目中开启一个新的 Issue
c_{opr}	Open Pull Request	提交一个新的 Pull Request
c_{prrc}	Pull Request Review Comment	参与到项目某一具体 Pull Request 下的讨论
c_{prm}	Pull Request Merged	提交的某一 PullRequest 被合入到项目代码仓库中

基于上述开源项目活跃度评测数据指标，我们定义某开发者在某项目上一段具体时间内的活跃度计算方式为：

$$A_d=\sum w_i c_i$$

其中的 A_d 为开发者活跃度，而 c_i 为上述五种行为事件由该开发者触发的发生次数，w_i 为该行为事件的加权比例。通过该公式，我们可以计算得到网络中开发者节点和项目节点之间连边的权重大小。

3.3. 显式关系建模

在开源贡献二分网络图中，如果开发者对项目产生过贡献，则开发者和项目之间即存在一条边。与

BiNE [15]等网络表示学习模型中所采用的一阶邻近度建模方法类似，我们通过考虑相互连接的开发者节点与项目节点之间的局部邻近度来建模显式关系。具体地：

开发者节点 u_i 和项目节点 v_j 之间的共现概率为：

$$P(i,j)=\frac{w_{ij}}{\sum_{e_{ij}\in E} w_{ij}}$$

其中 w_{ij} 为边 e_{ij} 的权重，公式表明，如果两个节点间连边的权重越大，即开发者与项目之间的联系越紧密，则它们共同出现的概率就越高。同时，我们采用重建分布的方法，用 u_i 和 v_j 嵌入向量的内积来定义相连接的两个节点之间的局部相似性，则开发者 u_i 和项目 v_j 之间的联合概率分布为：

$$\tilde{p}(i,j)=\frac{1}{1+\exp\left(-\vec{u_i}\cdot\vec{v_j}\right)}$$

为了在低维空间保持节点的显式关系，一种直接的方法是最小化这两种分布的目标函数。将两种分布之间的距离替换为 *KL* 散度，从而学习得到节点的表示向量，忽略无关约束后，得到：

$$O_1=KL\left(P\,\|\,\widehat{P}\right)=-\sum_{e_{ij}\in E} w_{ij}\ln\widehat{P}(i,j)$$

3.4. 隐式关系建模

在开源贡献二分网络中，相同类型的两个节点之间不存在直接相连的边。但如果相同类型的两个节点之间存在着路径，如图 2 中 David 和 Marry 都参与了 elasticsearch 项目的贡献，同时与该项目存在连接关系，则表明它们之间存在某种隐式关系，并且路径的数量和长度即表示隐式关系的强度[15]。因此我们除了对开发者节点和项目节点之间的显性关系建模，还考虑了相同类型的节点之间的隐性关系。

首先，将开源贡献二分网络转化成两个同质网络，即开发者网络和项目网络。根据 Co-HITS [16]，我们定义开发者网络的权重矩阵为：

$$w_{ij}^{U}=\sum_{k\in V} w_{ik}w_{jk}$$

上式表明两个开发者节点之间的关联程度由所有桥接这两个开发者的项目决定。同理可得项目网络的权重矩阵为：

$$w_{ij}^{V}=\sum_{k\in U} w_{ki}w_{kj}$$

因此我们可以使用对称矩阵 $W^U=\left[w_{ij}^U\right]$ 和 $W^V=\left[w_{ij}^V\right]$ 分别表示这两个同质网络的邻接矩阵。接着我们分别在这两个同构网络上执行有偏随机游走，将从每个节点开始的随机游走的数量与其中心度相关联，即节点的中心度越大，则从它开始的随机游走序列越多，最后得到开发者序列和项目序列两个语料库 D^U 和 D^V。

接着我们在这两个语料库上使用 skip-gram 模型，学习节点的表示向量。具体地，给出开发者语料库 D^U 中的一个节点游走序列 S，令 $C_S(u_i)$ 表示开发者节点 u_i 的上下文节点，得到以下目标函数：

$$\max_f O_2=\prod_{u_i\in S,S\in D^U}\prod_{u_c\in C_S(u_i)} P\left(u_c\middle|u_i;f\right)$$

同样，可以得到项目语料库 D^V 的目标函数为：

$$\max_f O_3=\prod_{u_j\in S,S\in D^V}\prod_{v_c\in C_S(v_j)} P\left(v_c\middle|v_j;f\right)$$

根据现有的网络嵌入方法，采用嵌入向量的内积和 Softmax 函数来表示中心节点 u_i (或 v_j)与它的上下文节点 u_c (或 v_c)的共现概率：

$$P\left(u_c|u_i\right)=\frac{\exp\left(\overrightarrow{u_i}^T\overrightarrow{\theta_c}\right)}{\sum_{k=1}^{|U|}\exp\left(\overrightarrow{u_i}^T\overrightarrow{\theta_k}\right)}$$

$$P\left(v_c|v_i\right)=\frac{\exp\left(\overrightarrow{v_j}^T\overrightarrow{\vartheta_c}\right)}{\sum_{k=1}^{|U|}\exp\left(\overrightarrow{v_j}^T\overrightarrow{\vartheta_k}\right)}$$

3.5. 融合显式关系与隐式关系的个性化推荐算法

得到了开源贡献网络的显式关系和隐式关系后，将它们的目标函数结合起来，建立联合优化框架：

$$O=\alpha\left(\ln O_2+\ln O_3\right)-\beta O_1$$

其中，α 和 β 为超参数，通过调节 α 和 β 的大小以适应不同的网络，如果 α 越大，则网络的隐式关系对节点表示学习的影响越大，β 越大，则表示显式关系的影响越大。

得到开发者节点和项目节点的嵌入表示 u_i 和 v_j 后。通过以下具体公式计算得到开发者和项目的相关度得分：

$$score\left(u_i,v_j\right)=\overrightarrow{u_i}^T\times\overrightarrow{v_j}$$

通过该公式，我们可以计算得到任意的某一开发者和某一项目的相关性得分。基于此，我们计算开发者与所有的候选推荐项目的得分，并根据分数排名得到 Top-N 的项目，最后将这些候选项目推荐给开发者。

4. 实验结果与分析

4.1. 实验数据集

本文采用 GHTorrent (2016/10/6)的公开数据集。GHTorrent 是一个离线的 GitHub 数据镜像，通过调用 GitHub REST API 接口来获取 GitHub 开放的公共事件流数据，归档记录 GitHub 平台所提供的问题讨论(Issue)、合并请求(Pull Request)等功能的用户历史行为数据，并以 Json 格式存储在 MongoDB 数据库中。在实验过程中，筛选出具有代表性的拥有至少 10 个仓库以上的 1621 位开发者，这些开发者每位都至少拥有 10 个仓库。对这些开发者的历史行为数据进行抽取，主要包括 issue、issue comment、pull request、pull request comment、pull request review comment 等，共涉及 20367 个项目。

4.2. 评价标准

考虑到本文推荐场景的特殊性，开发者由于时间和精力的限制，在一定时间内参与贡献的项目数量有限，因此我们选用正确率 Accuracy 指标来验证本文方法的有效性，具体定义公式如下所示：

$$\text{Accuracy}=\frac{\left|\left\{u|u\in U,R\left(u\right)\cap T\left(u\right)\neq\varnothing\right\}\right|}{|U|}$$

其中 U 表示测试集中的所有开发者集合，$R\left(u\right)$ 表示向开发者 u 推荐的项目集合，$T\left(u\right)$ 表示开发者 u 在测试集中实际参与贡献的项目集合。倘若 $R\left(u\right)$ 与 $T\left(u\right)$ 的交集不为空，则说明开发者实际至少参与了一个推荐的候选项目。

4.3. 实验结果与分析

我们选取了基于项目的协同过滤推荐、基于用户的协同过滤推荐、基于用户行为和项目特性的推荐三个对比模型来验证本文所提方法的性能：

基于项目的协同过滤推荐：该算法基于开发者对已参与项目的评分，通过计算项目之间的相似度，将评分最高的若干个相似项目推荐给开发者。

基于用户的协同过滤推荐：该算法首先计算开发者和开发者之间的相似度，找到与目标开发者特征相似的若干开发者，然后获取这些开发者所参与的项目集合，并将集合中的目标开发者未参与的项目作为推荐项目列表。

基于用户行为和项目特性的推荐：该算法首先从项目的描述文档中提取项目之间的相似度，并从开发者历史行为数据中提取开发者的行为特征，然后使用模拟退火算法自动优化参数配置，为开发者推荐与其最为相关的 Top-N 个项目。

不同的推荐模型实验结果如表 2 所示。实验结果表明，本文提出的方法在推荐 Top-5、Top-10、Top-15、Top-20 个项目时的表现均优于其他推荐模型，说明本文的模型能更好地提取开发者在开源贡献过程中所表现出来的显式和隐式的兴趣偏好，从而提高推荐的准确性。

Table 2. Experiment result table
表 2. 实验结果表

	项目协同过滤(%)	用户协同过滤(%)	基于项目内容(%)	本文的方法(%)
Top-5	4.37	12.68	18.61	29.85
Top-10	6.31	19.46	24.94	34.16
Top-15	7.68	23.54	28.22	38.84
Top-20	8.95	29.74	30.96	42.37

在此基础上，我们将 commit 参数引入到活跃度模型中，在计算开发者对项目的活跃度贡献值时，考虑开发者在项目仓库上的提交数，得到的推荐正确率如下图 3。

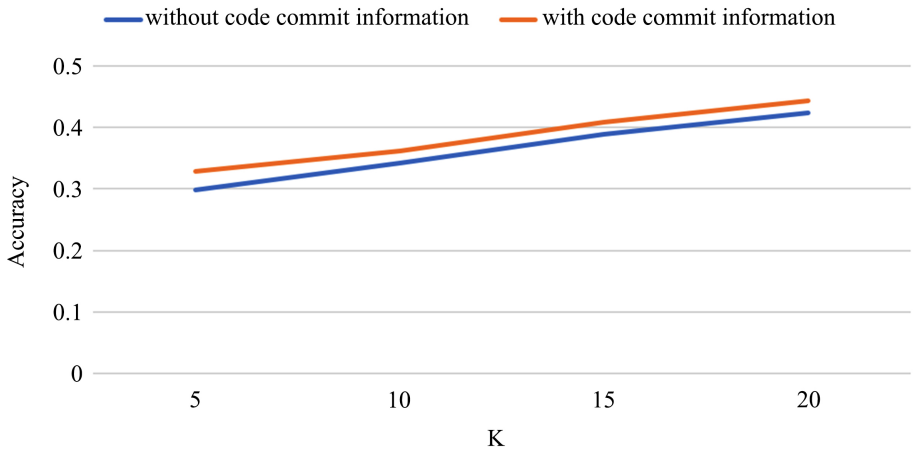

Figure 3. Comparison of recommendation performance after adding code commit information
图 3. 加入代码提交信息后的推荐结果比较

从图 3 中可以看出，加入 commit 参数后，推荐模型的性能得到了一定的提升，这是因为代码提交行为同样是开发者的开源贡献行为之一，是活跃度模型的补充。当模型越完整时，就越能体现开发者对项目的兴趣和参与开源贡献时的特征偏好，从而提高推荐模型的性能。

5. 总结与未来展望

本文针对开源场景下的项目推荐问题，提出了一种基于二分网络表示学习的开源项目推荐方法，基于活跃度模型将开发者参与项目贡献的过程建模成开源贡献二分网络，并融合开发者与项目之间的显式关系和隐式关系得到开发者的兴趣偏好，从而向开发者进行个性化开源项目推荐。最后在实验数据集上将本文的方法与传统的项目推荐方法进行比较，实验结果表明，本文所提的方法综合考虑了开发者与项目之间的显式关系和隐式关系，能更好地表征开发者的兴趣偏好，从而有效地提升的推荐效果。

未来的工作可以关注如何更好的建模开源协作场景下的复杂交互关系，从而更好地从开发者的历史贡献记录中捕捉到开发者隐式的兴趣偏好，从而提高项目推荐任务的性能。此外，为了更精确地评估本文所提方法的真实效果，考虑将推荐的结果发送给部分 GitHub 中的开发者，基于开发者真实的反馈进一步提高方法的准确性。

基金项目

国家自然科学基金(61772372)，上海市大数据管理系统工程研究中心开放课题。

参考文献

[1] Boyd, D.M. and Ellison, N.B. (2007) Social Network Sites: Definition, History, and Scholarship. *Journal of Computer-Mediated Communication*, **13**, 210-230. https://doi.org/10.1111/j.1083-6101.2007.00393.x

[2] Storey, M.A., Treude, C., van Deursen, A., *et al*. (2010) The Impact of Social Media on Software Engineering Practices and Tools. *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research*, Santa Fe, 7-8 November 2010, 359-364. https://doi.org/10.1145/1882362.1882435

[3] Begel, A., DeLine, R. and Zimmermann, T. (2010) Social Media for Software Engineering. *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research*, Santa Fe, 7-8 November 2010, 33-38. https://doi.org/10.1145/1882362.1882370

[4] Yang, C., Fan, Q., Wang, T., *et al*. (2019) RepoLike: A Multi-Feature-Based Personalized Recommendation Approach for Open-Source Repositories. *Frontiers of Information Technology & Electronic Engineering*, **20**, 222-237. https://doi.org/10.1631/FITEE.1700196

[5] Qing, Q., Jian, C. and Yancen, L. (2020) The Evolution of Software Ecosystem in GitHub. *Journal of Computer Research and Development*, **57**, 513-524.

[6] Franco-Bedoya, O., Ameller, D., Costal, D., *et al*. (2017) Open Source Software Ecosystems: A Systematic Mapping. *Information and Software Technology*, **91**, 160-185. https://doi.org/10.1016/j.infsof.2017.07.007

[7] Åhs, F. (2017) Evaluation of Memory Based Collaborative Filtering for Repository Recommendation on Github.

[8] Guendouz, M., Amine, A. and Hamou, R.M. (2015) Recommending Relevant Open Source Projects on Github Using a Collaborative-Filtering Technique. *International Journal of Open Source Software and Processes* (*IJOSSP*), **6**, 1-16. https://doi.org/10.4018/IJOSSP.2015010101

[9] 何锴琦, 马宇骁, 张炎, 刘华虓. 一种基于数据的 GitHub 项目个性化混合推荐方法[J]. 吉林大学学报(理学版), 2020, 58(6): 1399-1406.

[10] Sun, X., Xu, W., Xia, X., *et al*. (2018) Personalized Project Recommendation on GitHub. *Science China Information Sciences*, **61**, Article ID: 050106. https://doi.org/10.1007/s11432-017-9419-x

[11] Zhang, Y., Lo, D., Kochhar, P.S., *et al*. (2017) Detecting Similar Repositories on GitHub. 2017 *IEEE* 24*th International Conference on Software Analysis, Evolution and Reengineering* (*SANER*), Klagenfurt, 20-24 February 2017, 13-23. https://doi.org/10.1109/SANER.2017.7884605

[12] Yang, C., Fan, Q., Wang, T., *et al*. (2016) Repolike: Personal Repositories Recommendation in Social Coding Communities. *Proceedings of the* 8*th Asia-Pacific Symposium on Internetware*, Beijing, 18 September 2016, 54-62.

https://doi.org/10.1145/2993717.2993725

[13] Zhou, T., Wang, W. and Zhao, S. (2021) Open Source Galaxy: Heterogeneous Information Networks in Social Coding. 2021 *IEEE* 6*th International Conference on Big Data Analytics* (*ICBDA*), Xiamen, 5-8 March 2021, 349-355. https://doi.org/10.1109/ICBDA51983.2021.9403169

[14] 王伟, 周添一, 赵生宇, 范家宽. 全球开源生态发展现状研究[J]. 信息通信技术与政策, 2020(5): 38-44.

[15] Gao, M., Chen, L., He, X., *et al*. (2018) Bine: Bipartite Network Embedding. *The* 41*st International ACM SIGIR Conference on Research & Development in Information Retrieval*, Ann Arbor, 8-12 July 2018, 715-724. https://doi.org/10.1145/3209978.3209987

[16] Deng, H., Lyu, M.R. and King, I. (2009) A Generalized Co-Hits Algorithm and Its Application to Bipartite Graphs. *Proceedings of the* 15*th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Paris, 28 June-1 July 2009, 239-248. https://doi.org/10.1145/1557019.1557051