ISSN 2278-3091
Volume 10, No.5, September - October 2021
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse181052021.pdf
https://doi.org/10.30534/ijatcse/2021/181052021

Relevant Projectrecommendation System Using Developer Behavior And Project Features

Soohan Abbasi[1], Zulfiqar Ali[2] ,Rajesh Kumar[3], Madiha Shaikh[4], Komal Maheshwari[4],Paras Lal[5], Sagar Kumar[5]
[1,2,5]Department of Computer Science, FAST-NUCES Karachi,Pakistan.
Abbasi.soohan@gmail.com, zulfiqar.memon@nu.edu.pk, paraslal22@gmail.com
[3]Department of Computer Science, Beijing Institute of Technology,Beijing,China.
rajesh.kumar@hamdard.edu.pk
[4]Department of Software Engineering,Mehran University Jamshoro, Pakistan
madihashaikh162@gmail.com
[3,4]Department of Computing,Hamdrad University Karachi, Pakistan
Komal.maheshwari@hamdard.edu.pk
[5]Department of Computer Science & IT,NEDUET Karachi,Pakistan.
Sagarrathi099@gmail.com

Received Date : August 08, 2021 Accepted Date : September 16, 2021 Published Date : October 06, 2021

ABSTRACT

GitHub is a product improvement stage that advances participation and joint exertion in project advancement. Generally, engineers investigate related activities to reuse capacities and during investigating they coincidentally find a few elements that might help them in their undertaking. Recommending developers some projects that are similar to their work can save their time. Yet, it is difficult getting relevant projects amongst the pool of many projects on GitHub. Besides, every other user may have different requirements and choices for the project. A recommendation system saves developers from spending their time searching for projects that can help them in developing. In this paper, we propose an interactive and customized recommendation approach that recognizes software project features and developer behavior. This proposed approach naturally suggests the top-N most comparable programming projects. The outcomes utilizing information slithered from GitHub shows that our proposed approach suggest significant tasks with high review at cutoff 5 and 10.

Key words : Github, recommendation system, project behavior, user behavior, source code.

1. INTRODUCTION

The One of the leading class of machine learning algorithms is recommendation systems that propose related recommendations to users. Effectively, recommender systems contain a class of algorithms and techniques that can recommend “relevant” items to users. Spotify, YouTube, Netflix use Recommender systems and other several platforms use Recommender systems to create a playlist for video and music. Services like Amazon uses Recommender systems to recommend products, social media sites like twitter and Facebook use them for social media programs. Single inputs like music, or multiple inputs into the platforms or across the platforms such as books, news, and search queries. For explicit points like cafés and internet dating are accessible. At first, recommender frameworks were made to suggest research articles, writers, cooperation and extra security. The recommended items are as related to the user as possible so that the user can use those items. Recommended systems rank items according to their relevancy, then users are shown the most relevant items. The pertinence is something that the recommender framework should decide and is essentially founded on verifiable information. Importance of things depends on verifiable information of clients or things and pertinence is the factor that the suggested framework decides and do recommendations according to it.

There are so many projects on GitHub that gain less attention from developers because not all developers know about them [7]. On the other side, the same similar projects are initiated again and again by users which is a waste of time and efforts [13]. When developers search for similar projects on any search engine, the search engine only focuses on the matching of text rather than project similarity measure because it’s difficult to describe the project by choosing less number of keywords [1]. So all developers think that there should be a recommendation system which can recommend similar projects to developers and can also address these issues.

Over the time so many recommendation approaches are proposed but those approaches work only on project description and source code but do not consider user behavior as every user has different choice so the recommendation results are not accurate. As recommendation systems are single click system which means click, and then user’s need

are considered. We propose an interactive andcustomized recommendation approach that recognizes software projects features and developer behavior. To consider engineer practices in the suggestion, we dissect different activities, for example make, fork and star. To think about the components of the venture, we dissect and select terms from undertaking's portrayal. To suggest the top-N most related activities, the recommended approach combines developer behavior and project features.

There are two significant sorts of recommender frameworks content-based separating synergistic sifting. Recommender frameworks ordinarily utilize anybody or both. Collective separating approaches utilize comparative choices made by different clients and client's previous conduct and assemble a model on them. This model is utilized to anticipate the evaluations of various things or clients. It doesn't give us a rundown of any suggestion, though content-based sifting utilize side information on thing or client and furthermore think about the recorded conduct or evaluations of client.. There are a variety of techniques one can use to tune their recommendation according to their choices. It depends on the use case in use to get recommendations. The alternative to the search engine is the recommendation engine. One can achieve from the recommendation system as a search engine as they help the user to find the items they looking for and they couldn't have found them otherwise.

2. PROBLEM STATEMENT

GitHub is a product improvement stage that advances participation and joint exertion in project improvement. Normally, engineers investigate related ventures to reuse capacities and during investigating they coincidentally find a few elements that might help them in their undertaking. Suggesting designers a few activities that are like their work can save their time. However, it is troublesome getting important undertakings among the pool of many ventures on GitHub. In addition, each and every other client might have various necessities and decisions for the venture.

3. RELATED WORK

Personalized and interactive approach to recommend similar projects was proposed by Xiaobing et al [1] in which projects were recommended based on the actions on the projects in the repository of the user on GitHub. The approach was evaluated with the datasets of four different groups which represented 3 different areas of expertise and one mixed from GitHub. GitHub supports crowd based software engineering [2]. Ling Xiao Zhang et al proposed an approach which took the recorded data of user behavior from user repository and recommended open source projects. This approach uses the data of the projects of the 9 most common programming languages. Xin Xia [3] gave an approach which recommended top N projects by taking both the developers behavior and projects' feature on GitHub[4]. For developers' behavior the ratings like watch, comment, and history on the GitHub were considered and for project feature the description document of project and the source code document were analyzed[9].

Apache Spark spins around the idea of 'tough appropriated dataset (RDD)' 'which is a shortcoming open minded assortment of tuples with fixed formatting that can be processed in parallel' [3]. To realize the key point of their parallel algorithm they designed the RDD in 'Directed Acyclic Graph (DAG)' figure 2 shows the transformation of algorithm.

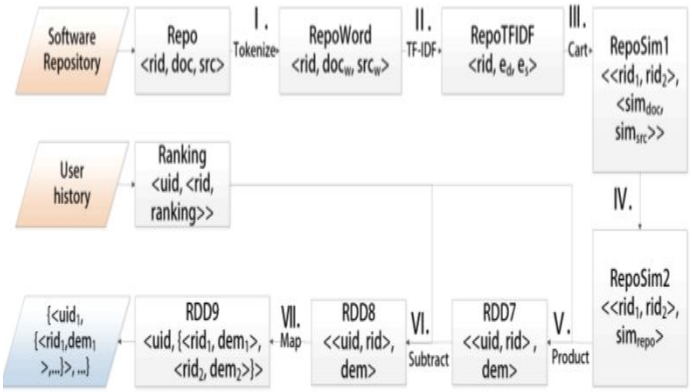

Figure 1: The process of RDD transformation.

4. METHODOLOGY

In this section, we have disccued the over view of architecture of our proposed solution. First from each repository we extractedproject descriptions and preprocessed the descriptions for later use. Then extracted features from preprocessed data. After getting feature vector of project description project profile was created. For user profile, user behavior was extracted and to overcome the user cold start problem the user data went through some processing. After getting the processed user data, user profile was created. Cosine similarity was calculated between user profile and project profile[11,12]. Then most relevant projects are recommended to the user.Setups required for this whole project was python 3.7 with jupyter notebook and some python libraries to work with. Figure 14 shows the architecture of proposed approach.

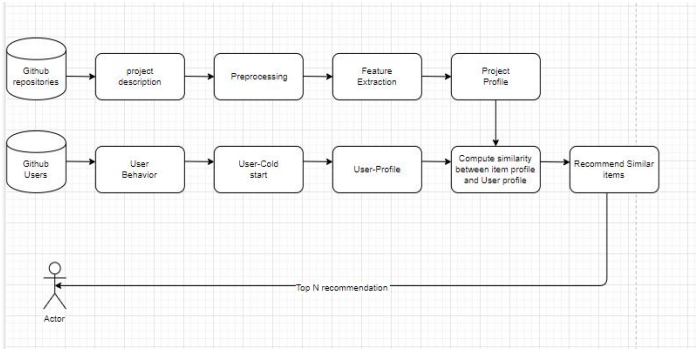

Figure 2: Overview of proposed dataset

We worked with a crawler to fetch and sort user behavior and repositories due to the restricted usage number of the GitHub API. These projects were extracted from different

organizations on GitHub. Our projects are of different programming languages. Our dataset consists of two CSV files. One file has projects and their ids and the other file has users with their ids and their different behaviors. We 1st extracted projects and then extracted all the users who have performed some actions on those projects along with their actions (user behavior). So, the number of projects we extracted is 6573 and the number of users we extracted is 3097. After preprocessing done on user data, we randomly divided user data into training and test set data with the ratio of 80% and 20%. Note that CF1, Popularity filtering2, and the proposed approach utilized a similar preparing and test sets; be that as it may, the proposed approach utilized venture content, which CF and PF couldn't utilize. Following figures show our dataset. Below Figure shows the overview of dataset.

	Project_id	Repo_description
0	26	Grit gives you object oriented read/write acce...
1	27	Chronic is a pure Ruby natural language da...
2	28	Audited (formerly acts_as_audited) is an ORM e...
3	29	A Ruby interface to the Twitter API.
4	31	Annotate Rails classes with schema and routes ...
5	35	AASM - State machines for Ruby classes (pl...
6	36	mini replacement for RMagick
7	42	A tagging plugin for Rails applications that a...
8	43	Sequel: The Database Toolkit for Ruby
9	48	Beanstalk is a simple, fast work queue.

Projects Dataset

	User_id	Project_id	Behaviour
0	4	26	Fork
1	4	26	CREATE
2	1	26	STAR
3	17	27	Fork
4	4	27	CREATE
5	2	35	STAR
6	2	36	Fork
7	21	42	CREATE
8	21	43	STAR
9	25	48	Fork

User dataset

Figure 3: Research Methodoly of System

To make a prediction, personalized projects take advantage of a user's history by considering the preferences and likes to build a user profile. The activities and behavior they perform on users, by considering those activities and preferences their user profile is made. The problem arises when a new user or item enters the system and the system does not have any information about that new item/user or has a minimum amount of information. This is called a cold start problem. This word is derived from cars. When there is extreme cold the vehicles take a lot of time to get started the reason is that they don't get their optimal temperature to function correctly. But once they get their optimal temperature they will run smoothly. The cold start simply means that the surrounded circumstances are not fully in favor to work properly and give the best optimal results. To overcome this problem we have kept only those users who have made at least five interactions.
The relevant description depends on the varietyof assignments to be accomplished, the text-based system needs any description of documents. Furthermore, the capacity to. Correctly execute a group task depends on the descriptionof documents to be classified. Modified fromdata mining that manages the well-structured text mining data and contracts with several even unstructured, semi-structured documents. This executes that one of the principal points[14]. To analyze accurately several algorithms and hyper parameter options for models, evaluation is essential for machine learning projects. The most important feature of evaluation is to make sure that the trained model postulates for the independent dataset using cross-validation. We have used a simple technique of cross-validation called handout which randomly split the dataset into an 80% training dataset and a 20% test set. We have computed all the evaluations using the test set.
There are a set metrics commonly used for evaluation in the recommendation system. We have chosen Top-N accuracy metrics to work with and evaluate our recommendation system. These metrics evaluate the top N recommendations provided to the user. We have chosen to go with recall at cutoff N and hit rate cutoff N. According to3,4 in recommendation system recall is the rate of relevant items selected out of all the relevant items. Formula to calculate the recall is shown in figure.

$$\text{recommender system recall:} \quad r = \frac{\#\text{ of our recommendations that are relevant}}{\#\text{ of all the possible relevant items}}$$

Figure 4: Formula to calculate recall rate

5. RESULTS

We utilized the Top N exactness metric to contrast the proposed approach with CF and PF. The equation to register review for suggestion frameworks is given in the assessment segment. The observational outcomes are displayed in the table underneath. The outcomes show that the review at remove 5 of the proposed approach is a lot higher than those two. If we see close to the results of CF and proposed approach we can analyze that CF has recommended 0 similar items at cutoff 5. CF did not consider user features of the project so did the popularity matrix. If we analyze the result of 3 models at the cut off 10 we can see that the proposed approach has a higher recall, which means that there were more relevant items. The hit rate of the proposed approach at both cut off five relatively high rates compared to the other

1

two. Table shows the Comparison of results of different approaches and proposed approach.

Table 1: Shows the Empirical results of different approaches

Approaches	Recall@5	Recall@10
Proposed approach	88%	93%
Popularity Based Recommendation	13%	26%
Collaborative Filtering Recommendation	0.0%	0.002%

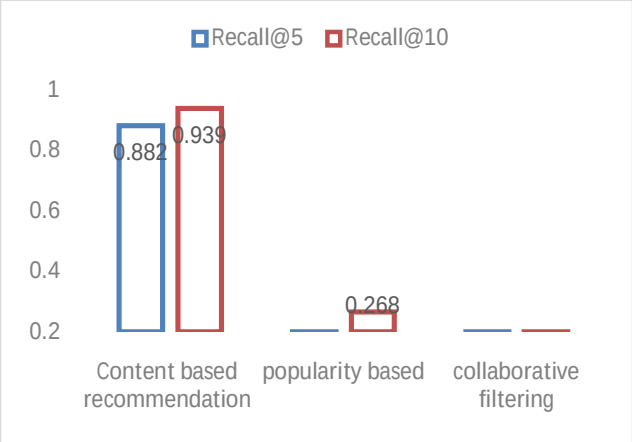

Figure 5: The bar chart for different approached

6. Conclusion

GitHub is a product improvement stage that advances participation and joint exertion in project advancement. Normally, developers explore related projects to reuse functions and during exploring they stumble upon some features that may help them in their project. Recommending developers some projects that are similar to their work can save their time. In this paper we have proposed an approach which recommends the top N similarity projects to a user. This approach is performed on the basis of user behavior and project features. We extracted project features and compute similarity with user profile to know the preferences of user. Then we recommended the most similar projects to the user. We compared our approaches with different approaches for recommending similar projects and we saw that the different approaches cannot propose similar recommendations to the user based on the item profile. We chose two complete different approached in which one work with only the history of items but not with the users. We saw that if we only consider the history of items, user cannot get the desired result. The other approach we compared our approach is popularity based approach in which user get only those items which has high rate rank in rank list.We see how our approach outperformed the other two approaches in recommending similar projects because the proposed approach considers both users and items. It take user's preferences from user profile and similar items from item history. We got recall of 83% at cutoff 5 and 93% recall at cutoff 10 It help to save time to check copies manually.

REFERENCES

[1] Xiaobing SUN1, 2, 5*, Wenyuan XU1, Xin XIA3, Xiang CHEN4 & Bin Li, "Personalized project recommendation on GitHub", ACM, Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering, 2018.

[2] Lingxiao Zhang, Yanzhen Zou, "Recommending Relevant Projects via User Behaviour: An Exploratory Study on Github", ACM, Proceedings of the 1st International Workshop on Crowd-based Software Development Methods and Technologies, 2014

[3] Xin Xia., Scalable Relevant Project Recommendation on GitHub", Proceedings of the 1st International Workshop on Crowd-based Software Development Methods and Technologies, 2017, Pages 25-30

[4] Ferdian THUNG, ferdiant, David LO, Julia LAWALL, "Automated Library Recommendation", 20th Working Conference on Reverse Engineering (WCRE), 2013

[5] A. Felfernig& G. Ninaus, "RepoLike: a multi-feature-based personalized recommendation approach for open source repositories", Proceedings of the 13th International Conference on Mining Software Repositories, Pages 500-503, 2016

[6] XiaobingSun, BinLi, YucongDuan, WeiShi, andXiangyueLiu, "Mining Software Repositories for Automatic Interface Recommendation", Scientific Programming archive, Volume 2016, June 2016

[7] FragkiskosChatziasimidis&IoannisStamelos, "REPERSP: Recommending Personalized Software Projects on GitHub", IEEE International Conference on Software Maintenance and Evolution (ICSME), 2017

[8] CHAO LIU1,2, DAN YANG2, XIAOHONG ZHANG2, BAISHAKHI RAY3, AND MD MASUDUR RAHMAN4, "Recommending GitHub Projects for Developer Onboarding", ACM, Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings, 2018, Pages 319-320

[9] Yun Zhang1, David Lo2, Pavneet Singh Kochhar2, Xin Xia1‡, Quanlai Li3, and Jianling Sun1, "Detecting Similar Repositories on GitHub" IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER). Doi: 10.1109/saner, 2017

[10] TadejMatek∗ and SvitTimejZebec, "GitHub open source project recommendation system", IEEE-ACCESS, 2016.

[11] Yue Yu∗, Huaimin Wang∗, Gang Yin∗, Charles X. Ling, "Reviewer Recommender of Pull-Requests in GitHub", IEEE International Conference on Software Maintenance and Evolution, 2014.

[12] Laura Dabbish, Colleen Stuart, Jason Tsay, Jim Herbsleb, "Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository", Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work, Pages 1277-1286.

[13] MotaharehBahramiZanjani, HuzefaKagdi, Christian Bird, "Reviewer Recommender of Pull-Requests in GitHub", IEEE Transactions on Software Engineering, DOI 10.1109/TSE.2015.2500238, 2014

[14] Hans-Jörg Happel, Walid Maalej "Potentials and Challenges of Recommendation Systems for Software Development", Proceedings of the 2016 international workshop on Recommendation systems for software engineering,Pages 11-15