Pogo: A Programming Project Advisor

Maneet G, Mansi M, Neetha R, Aditi M[1], Vijaysri L, Harsh K[1]

I. PROBLEM DEFINITION

In today's world of constantly evolving technology, choosing the right programming tools to build a project can be daunting. Users need to google, search Stack OverFlow (SO), GitHub etc. separately to implement programming projects. With Pogo, users have a one-stop application which speeds up their learning.

To build Pogo, we fetch SO, Libraries.io and GitHub data from Google BigQuery and integrate them using topic modeling and graph analytics. On a user's programming task query, Pogo offers insights into most relevant technologies. Each technology is accompanied with information depicting its popularity, degree of assistance and any learning dependency it might have. With these insights users can make informed decisions on technology best suited for them. Pogo thereby reduces time and effort spent in the planning and learning phase of a project. Hence, people from both academia and industry would find the tool beneficial.

II. LITERATURE SURVEY

A. StackOverflow

A recent research established that trends in SO data are strongly correlated to trends in Google searches [1]. Based on this, we can rely on SO data to calculate popularity and assistance metrics of various technologies.

Latent Dirichlet Allocation (LDA) helps in figuring out themes describing the scope of a document [2], [23]. Integral to Pogo, LDA shall map our users' query to relevant SO entries. Linares-Vasquez et al. reported its use for figuring out hot topics related to mobile development [3]. One of our RQ is greatly aligned with this study. However, we plan to generate more accurate results over a larger dataset by using LDA-GA to generate a near-optimal model input configuration rather than adopting the baseline one. LDA-GA takes into account documents' cohesion and separation for describing a configuration's fitness, i.e., the Silhoutte Index [4].

Asaduzzaman explores the characteristics of unanswered questions and offers a qualitative criteria for their classification [5]. It can help us refine our parameters that rely on the total number of questions. However, it's not as useful since authors don't provide a quantitative way of achieving this.

Further, current practices also suggest that we might be able to exploit island parsing for fragment extraction out of SO discussions to help gain a better knowledge of word and fragment frequency [6]. And, to better understand the relations between question categories and problem categories we can implement kNN, albeit with limited accuracy [7]. We can also utilize this algorithm to generate synonyms for tags which might help categorize tags more efficiently [8].

Research done on SO tags also suggests that a discriminative model via Supporting Vector Machines (SVM) can be used to suggest, or in our case predict tags based on existing ones [9]. Using SO for predicting most successful answers to a user query, we can use sentiment analysis and logistic regression to get best results [10]. Also, in certain cases, for instance, while predicting the most popular posts, we must be thorough and not limit ourselves to the initial content [11].

B. GitHub

Another RQ is aligned with determining popularity of repositories. Hudson Borges [14] discusses factors that contribute to higher star count on a GitHub project and identifies a pattern of popularity growth by clustering time series data. In a similar study, regression analysis was performed to model how folder usage relates to project popularity [15]. Current practices discretize data using

College of Computing, [1]College of Engineering, Georgia Institute of Technology, Atlanta, GA 30318, USA

k-means and measure success rate by cardinality of downloads [16]. This gives an interesting pattern associating open source projects and owner features with the success rate. As proposed by Cai [17], we plan to employ a graph based approach (GRETA) to assigning tags for repositories using domain knowledge from SO. This would allow GitHub repositories to be efficiently accessed and understood.

Detecting repositories that are similar to each other is integral to Pogo. Research suggests user's starring repositories and repository read-me files are good indicators of similarities [22]. As proposed by Dabbish, we can make a rich set of social inferences from networked activity information in GitHub, such as guessing which of the similar projects have the best long-term chance of thriving [18]. Combining multiple regression modeling with visualization and text analytics helps suggest which programming language can achieve the best software quality [19]. Sometimes data can be easily misunderstood. Recommendations exist on how to best use data available from GitHub and avoid analysis risks [20].

C. *Neo4j*

Representation and efficient querying of interconnected data are key challenges to address, while building an effective, interactive tool. The paper representing time-varying social network data as a property graph in Neo4j database produces good performance results in querying, exploratory data analysis, and research-oriented data mining [12]. Studies have reported empirical comparison between two graph query languages to access data in Neo4j, Cypher and Gremlin [13], [21]. Results show Neo4j to be a high-performance replacement for relational databases, especially when handling highly interconnected data. However, unlike the study, we plan to use Cypher with Neo4j (v1.9 or above) for better performance.

III. INNOVATION

Integrating data from SO, GitHub and Library.io together in the form of a learning aid.

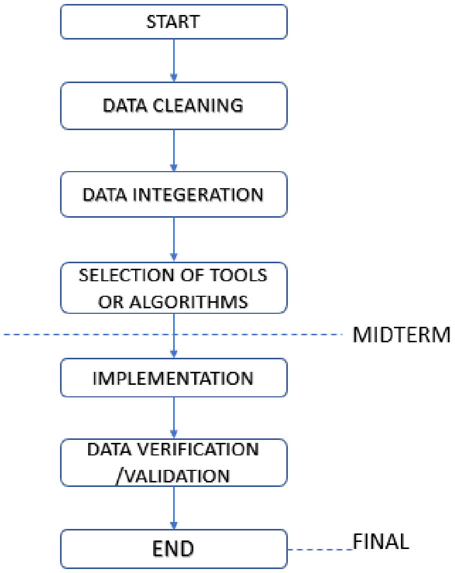
Such a tool is not yet available based on our survey. If successful, Pogo can bring down learning-curves for new programmers, and enhance productivity for experienced ones. The performance of this tool can be measured via user feedback, and the number of pageviews.

IV. HEILMEIER QUESTIONS

1) Q1-Q4: Section I.
2) Q5: Section III.
3) Q6-Q7: Section VI.
4) Q8-Q9: Section V.

V. PLAN OF ACTIVITIES

The work involved in building Pogo is evenly distributed among all six members in the team.



VI. RISKS, PAYOFF AND ASSOCIATED COSTS

The overwhelming amount of data that is available can pose scalability issues. Having a small proportion of missing data could also pose a risk. However, the payoffs include reduced learning time and boosted productivity for the user. Furthermore, the monthly costs associated are minimal for using Google BigQuery since the first 10GB of storage and 1TB of queries is free. A monthly cost of 0.02 USD/GB for storage and 5 USD/TB is incurred post this limit.

REFERENCES

[1] Chen, C., & Xing, Z., 2016, Towards Correlating Search on Google and Asking on Stack Overflow. In proceedings of COMPSAC, the IEEE Computer Society's International Computer Software & Applications Conference (pp 83-92).

[2] Blei, D.M., 2012. Probabilistic topic models. Communications of the ACM, 55(4), pp.77-84.

[3] Linares-Vsquez, M., Dit, B. and Poshyvanyk, D., 2013, May. An exploratory analysis of mobile development issues using stack overflow. In Proceedings of the 10th Working Conference on Mining Software Repositories (pp. 93-96). IEEE Press.

[4] Panichella, A., Dit, B., Oliveto, R., Di Penta, M., Poshyvanyk, D. and De Lucia, A., 2013, May. How to effectively use topic models for software engineering tasks? an approach based on genetic algorithms. In Proceedings of the 2013 International Conference on Software Engineering (pp. 522-531). IEEE Press.

[5] Asaduzzaman, M., Mashiyat, A.S., Roy, C.K. and Schneider, K.A., 2013, May. Answering questions about unanswered questions of stack overflow. In Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on (pp. 97-100). IEEE.

[6] Ponzanelli, Luca, Andrea Mocci, and Michele Lanza. ”StORMeD: Stack Overflow ready-made data.” Mining Software Repositories (MSR), 2015 IEEE/ACM 12th Working Conference on. IEEE, 2015.

[7] Beyer, Stefanie, and Martin Pinzger. ”A manual categorization of android app development issues on stack overflow.” Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on. IEEE, 2014.

[8] Beyer, Stefanie, and Martin Pinzger. ”Synonym suggestion for tags on stack overflow.” Proceedings of the 2015 IEEE 23rd International Conference on Program Comprehension. IEEE Press, 2015.

[9] Saha, Avigit K., Ripon K. Saha, and Kevin A. Schneider. ”A discriminative model approach for suggesting tags automatically for stack overflow questions.” Proceedings of the 10th Working Conference on Mining Software Repositories. IEEE Press, 2013.

[10] Calefato, Fabio, et al. ”Mining successful answers in stack overflow.” Mining Software Repositories (MSR), 2015 IEEE/ACM 12th Working Conference on. IEEE, 2015.

[11] Phukan, Devaraj, and Aayush Kumar Singha. ”Feasibility analysis for popularity prediction of stack exchange posts based on its initial content.” Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference on. IEEE, 2016.

[12] Ciro Cattuto, Marco Quaggiotto, Andre Panisson, Alex Averbuch. ”Time-varying Social Networks in a Graph Database: a Neo4j use case.” GRADES ’13 First International Workshop on Graph Data Management Experiences and Systems. ACM, 2013.

[13] Florian Holzschuher, Rene Peinl. ”Performance of graph query languages:comparison of cypher, gremlin and native access in Neo4j”. Extending Database Technology (EDBT) ’13 Proceedings of the Joint EDBT/International Conference on Database Theory (ICDT) 2013 Workshops. ACM, 2013.

[14] Hudson Borges, Andre Hora, Marco Tulio Valente. ”Understanding the Factors That Impact the Popularity of GitHub Repositories”. Software Maintenance and Evolution (ICSME), 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME),2016.

[15] Jiaxin Hu, Minghui Zhou, Audris Mockus. ”Patterns of folder use and project popularity: a case study of github repositories”.ESEM ’14 Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement,2014.

[16] Fragkiskos Chatziasimidis, Ioannis Stamelos. ”Data collection and analysis of GitHub repositories and users”.Information, Intelligence, Systems and Applications (IISA), 2015 6th International Conference on,2015.

[17] Xuyang Cai, Jiangang Zhu,Beijun Shen. ”GRETA: Graph-Based Tag Assignment for GitHub Repositories”.Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual,2016.

[18] Dabbish, L., Stuart, C., Tsay, J. and Herbsleb, J., 2012, February. Social coding in GitHub: transparency and collaboration in an open software repository. In Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work (pp. 1277-1286). ACM.

[19] Ray, B., Posnett, D., Filkov, V. and Devanbu, P., 2014, November. A large scale study of programming languages and code quality in github. In Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (pp. 155-165). ACM.

[20] Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D.M. and Damian, D., 2014, May. The promises and perils of mining github. In Proceedings of the 11th working conference on mining software repositories (pp. 92-101). ACM.

[21] Huang, H. and Dong, Z., 2013, November. Research on architecture and query performance based on distributed graph database Neo4j. In Consumer Electronics, Communications and Networks (CECNet), 2013 3rd International Conference on (pp. 533-536). IEEE.

[22] Zhang, Y., Lo, D., Kochhar, P.S., Xia, X., Li, Q. and Sun, J., 2017, February. Detecting similar repositories on GitHub. In Software Analysis, Evolution and Reengineering (SANER), 2017 IEEE 24th International Conference on (pp. 13-23). IEEE.

[23] Jie Zou, Ling Xu, Weikang Guo, Meng Yan, Dan Yang, Xiaohong Zhang. ”Which Non-functional Requirements Do Developers Focus On? An Empirical Study on Stack Overflow Using Topic Analysis.” Mining Software Repositories (MSR), 2015 IEEE/ACM 12th Working Conference. IEEE, 2015.