Evaluating Visual Explanation of Bug Report Assignment Recommendations

Shayla Azad Bhyan and John Anvik
Department of Mathematics and Computer Science
University of Lethbridge, Alberta, CANADA
E-mail: [s.bhuyan, john.anvik]@uleth.ca

Abstract

Software development projects typically use an issue tracking system where the project members and users can either report faults or request additional features. Each of these reports needs to be triaged to determine such things as the priority of the report or which developers should be assigned to resolve the report. To assist a triager with report assigning, an assignment recommender has been suggested as a means of improving the process. However, proposed assignment recommenders typically present a list of developer names without an explanation of the rationale. This work presents the results of a small user study to validate our approach to visually explaining bug report assignments.

1. Introduction

As the need for global and distributed software projects grows, so does the need for finding people with the required expertise for a given task. Recommendation systems have been proposed as a means for improving the achievement of this goal [1–4]. The typical recommendation system provides a textual list of recommendations with no explanation for why each recommendation was made. As described by Herlocker et al. [5], most current recommendation systems are a black box where transparency is not ensured.

Providing transparency by incorporating the reasoning and data behind a recommendation is an important feature of an effective recommendation system [1, 6], as recent work in "Explainable Artificial Intelligence" shows [7,8]. Effective visualizations can help to provide this transparency for a recommender system that uses multidimensional data and improve user acceptance rates for recommendations. Making efficient visualization of the recommendations can show how different dimensions were applied in making a recommendation to improve transparency [6], as well as improving a user's acceptance rate of recommendations [1]. Trintarev et al. [9] surveyed a group of movie-goers and found that the explanations behind the recommendations are as important to users as the recommendations.

Bug report[1] triage recommenders are an example of such a recommender system in software engineering. Bug report triage is the process where a project member, typically a project manager, decides what to do with a bug report. When projects receive many bug reports every day, bug report triage becomes a significant software maintenance issue [10–12]. Also, bug report triage is a tedious task that often shifts development resources away from improving a product to instead managing the project. Within the area of bug report triage recommenders, assignment recommenders are the most commonly researched (e.g. [4, 10, 13]). Typically, proposed assignment recommenders provide a textual list of recommended developers' names (e.g. [10, 14, 15]). Despite years of research, assignment recommenders have yet to be meaningfully integrated into products such as Bugzilla, GitHub and Jira. Prior studies (e.g. [16]) and informal discussions with developers indicate that one of the barriers to the adoption of such a system is the lack of explanation, leading developers to question and perhaps not trust the recommendations. It is these discussions that motivated this work.

This paper presents an evaluation of our initial work towards providing transparency for bug report triage assignment recommendations using visual explanations. We explore the use of stacked horizontal bars,

This work is supported by Alberta Innovates and the Natural Sciences and Engineering Research Council of Canada.
DOI reference number: 10.18293/SEKE2021-054

[1]We use the term "bug report" to refer to items in a project's issue tracking system.

a pie chart, and a data table. To assess the impact of the use of these visualizations, we conducted a small user study.

To the best of our knowledge, this area of bug report assignment recommenders has not been explored in the literature. We believe that part of the reason for this is that these recommenders are created typically using machine learning algorithms that make it hard to provide explanations. For example, the two most commonly used algorithms are Support Vector Machines (SVM) [17, 18] and the Random Forest algorithm [19, 20] where determining the rationale for the recommendations is near impossible. Instead, we focused on the use of Multinomial Na¨ıve Bayes and Topic Modelling, both of which use probabilistic models. The use of a probabilistic model makes for an easier determination of recommendation rationale. In contrast to Multinomial Na¨ıve Bayes which has been commonly used in the past, the use of Topic Modeling in this area is relatively new.

2. Visualization of Assignment Recommendations

To provide an assignment recommendation for a new bug report, first, the report is turned into a vector of features. Next, the features of the new bug report are given to the trained classifier. In the case of Multinomial Naïve Bayes, for each potential developer, the set of features that are common between their instances and the new bug report are collected. Then the sum of the conditional probability of each of these features is determined to represent the expertise score of that developer for that bug report. Finally, developers are ranked based on expertise scores. In most cases, bug reports have a lot of relevant words in common. If all of these words were displayed, the graphs would have too much information and be hard to read and understand. Therefore, we chose to display only the most relevant terms based on the TF-IDF score. We empirically found that providing more than five terms did not significantly improve the accuracy.

When choosing the visualizations to explore, we focused on simplicity and familiarity to potential users. Therefore, we chose the data-table format, stacked bar chart and pie chart, as they are commonly used in a variety of applications and are familiar to a wide range of people. Also, we sought visualizations that would allow for the display of information about relative contributions. In our case, that means the individual probabilities of terms or the dominant topic in a bug report will have towards the ranking of developers. Finally, these forms of visualization have been previously used in similar contexts [1, 21, 22].

Figure 1 shows these three types of visual representation for an assignment recommendation using Multinomial Naïve Bayes. The pie chart presents the important features from each report based on their conditional probability values. If a user clicks on the pie, a new web page opens. This new page shows a pie for each developer and each pie shows the overall conditional probability values for the corresponding recommended developer. The data table shows these same values for each important feature for each developer. The stacked bar chart however shows developers horizontally where each feature is represented by a different colour. The developer who has the highest sum of conditional probability values for all of the selected features is shown at the bottom.[2]

For the Topic Modelling classifier, the cluster with the shortest distance to the new bug report is determined and the ranked list of developers for that cluster forms the recommendation list. The pie chart shows the recommended developer names and their solved bug report rate for a specific topic. The data table gives the list of developer names with the exact number of reports that the developer solved related to that topic. The stacked bar chart also shows the developer names horizontally with their score. The colour of the bar is related to the selected topic.

3. Evaluation

Our empirical study[3] sought answers to three research questions. First, do developers find visual explanations of assignment recommendations easy to understand? Second, do developers trust visual explanations of assignment recommendations? Lastly, which of the three investigated visualizations is preferred?

The web application used in our study consisted of two parts: a web browser plug-in and a web service. To present a subject with visual explanations for the assignment recommendations, we created a web browser plug-in for Google Chrome [4].

To use the plug-in, first, a user opens a bug report in the web browser from a Bugzilla server. We configured the plug-in to only work with bug reports from Mozilla projects (i.e. those with the URL `https://bugzilla.mozilla.org`), as that was our chosen dataset. Next, the user clicks on a button

[2]That the top recommendation is shown at the bottom is a result of the graphics library used, not an intentional choice.

[3]An analytical evaluation of the underlying recommenders was conducted before the study. See [23] for these details.

[4]https://chrome.google.com/webstore/detail/recommend-expertise/clpcpddhohohhfcnkiknfopaeikbngid

labelled "Recommend Experts" in the plug-in in the browser. This makes a request to the web service with the bug report's id and opens a new browser window containing the response from the web service - an HTML page showing the assignment recommendations in a visual form. Figure 1 shows one of the four visualization web pages that are returned by the web service.[5]

When given the bug report id, the web service queries the issue tracking system for the title and description of the requested report. Stop words are removed and stemming applied to the text before being passed to a classifier. The results from the classifier are then used to create the visualizations. As previously mentioned, only the top five (5) recommended developers are shown to avoid information overload.

3.1. User Study

The user study consisted of a within-subject study where all participants received treatment. Our user study[6] consisted of three parts: a demographic survey, presentation of the visualizations with an accompanying survey, and a post-usage survey.

The demographic and post-usage survey was conducted using Qualtrics, and the visualization survey integrated into the web pages was generated by the web service. Participants were asked to complete the demographic survey first, then install the browser plug-in and go through the list of bug reports, and then complete the post-usage survey.

To recruit participants for our study we posted on Reddit in channels like `r/learnmachinelearning` and `r/AskComputerScience`. The criteria for participation was to either be in a two-year computer science post-graduate degree (i.e. in an M.Sc.-like program) or have more than one year of software development experience. Interested participants were asked to contact the primary researcher for a study id and further instructions. We were able to recruit fourteen participants.

As previously noted, this research direction is new in the software engineering area. Participants could have been recruited from the Bugzilla project (i.e. the data set used for training the assignment recommender), but we chose to conduct a small study first to assess the viability of our approach before approaching specific project developers. In other words, the purpose of the user study was to gain a general understanding of the effectiveness of visually representing bug report assignment recommendations. By having participants that were not associated with the particular project for which the assignment recommender was created, we sought to determine a base case for future investigations in this area.

To assess the effectiveness of the visualizations, each participant was given the same set of fifteen (15) links to pre-selected bug reports for the `Bugzilla` software product. The selected bug reports were randomly chosen from those that had a status of `Open` (i.e. not `Resolved`). This was done so that the reports reflected the general level of difficulty of reports present in the issue tracking system for the product (i.e. no consideration was given for the complexity of the bug report in their selection) and so that participants were not biased towards the recommendations by examining "the correct answer" of who should have been recommended as the assignee.

After clicking on a link for a bug report, the participant was taken to the actual bug report in the Mozilla project's issue tracking system. The participant would then click "Recommend Experts" in the plug-in and the web service would provide the recommendations as part of one of four randomly selected web pages. The participant would also be asked one of two sets of questions depending on the presented visualizations.

The intent of two of the web pages was to present participants with a single type of visualization (stacked bar or pie chart) with data from each of the recommenders. In this way, we could determine if participants preferred the use of one visualization approach over another. The intent of the other two web pages was to determine if participants preferred a particular type of classifier.

For the web pages that presented results from the two different classifiers (Multinomial Naïve Bayes and Topic Modelling), participants were asked several questions: Did they think the visualizations increased their understanding of the recommendation? Did they trust the recommendations? Did they think the visualizations provided enough information? If not, what visualization did they think was missing?

For the web pages where the results from the same classifier were presented, but the visualization differed (i.e. bar vs. pie vs. table), the participants were asked similar questions as before. Did they trust the recommendations? Did they think the visualizations provided enough information? If not, what visualization did they think was missing?

After participants finished using the browser plug-in on the fifteen bug reports, or however many they chose to do, they were asked to complete the post-usage survey. This survey asked their thoughts about our approach to providing visual explanations of bug report

[5]Examples of the other visualizations can be found in [23].

[6]The study was reviewed by the University of Lethbridge Ethics Committee and assigned protocol number #2019-070.

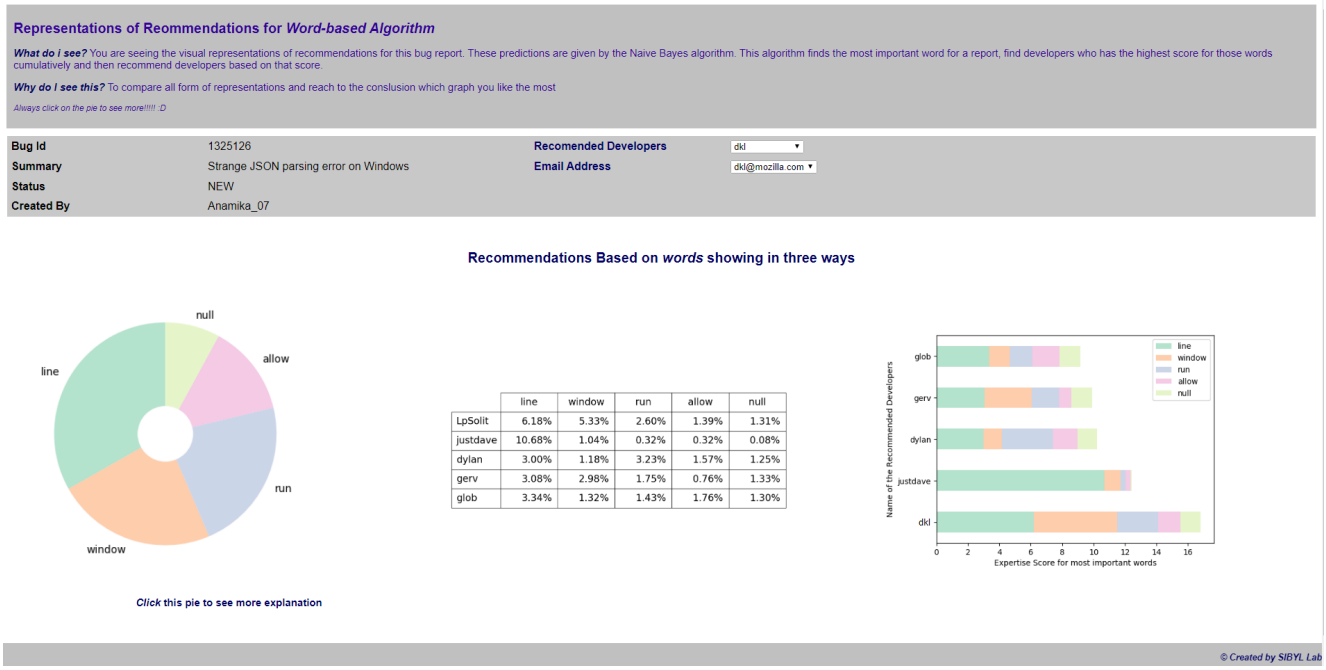

Figure 1. Visualization of recommendations using Multinomial Naïve Bayes classifier.

assignment recommendations. Examples of questions asked included: How important is the visual explanation of the recommendations to you? How would you improve the explanation of the recommendations? Did you think that one visualization was enough? Which combination of visualizations would you want for explaining an assignment recommendation?

4. Results

We found that most participants took an hour to complete the study, although one participant took much longer (2.5 hours) [7]

The occupations reported by the participants were: student (3), quality analyst (3), application developer (5), project manager (1), and application architect (1). The participants identified as 64% male and 36% female, and just over half (57%) of the participants had a graduate degree (Masters or Ph.D.). Participants' development experience varied from less than three years (1), four to nine years (7), and more than nine years (6). Most of the participants (71%) reported having logged a bug report, which indicates that most of them had some form of first-hand knowledge of how bug report assignment works. When asked about their level of familiarity with machine learning, two (2) reported themselves as beginners, and the rest considering themselves to have advanced knowledge.

4.1. Visualisation of Assignment Recommendations

Table 1 shows the results for the questions where we were trying to determine if there was a preference for one visualization over another. We can see that there was a slight preference for the stacked bar chart over the pie chart. We can also see that more than 70% of participants felt that these visualizations provided enough information. There was no notable difference in the preferred visualizations for developers with different experience levels. This may be a result of the participants not being intimately familiar with the project. Note that no participants preferred the data table over the other two, with one participant commenting "it is not interesting."

Regarding trust in the recommendations, we found that trust in both the Topic Modelling classifier and the Multinomial Naïve Bayes classifier was high, at 94% and 85%, respectively.

Participants felt that these visualizations provided them with enough information to make an informed decision (Multinomial Naïve Bayes – 97%, Topic Modelling – 100%). Table 2 shows that for both types of classifiers, most of the participants preferred the stacked bar chart over the other two data representations. As might be expected, participants preferred

[7] When asked, the participant responded that this was due to interest in the approach and wanting to fully understand it.

Table 1. Visualization Preference.

Question	Stacked Bar	Pie Chart
Do you think these visualizations increase your understanding of the recommendation?	77.00%	76.09%
How much do you trust these recommendations? (1 being not trustworthy at all to 5 being you trust this fully.)	3.46	3.41
Do you think these visualizations provide you enough information?	79.00%	77.78%

Table 2. Preference for specific visualizations.

Chart Type	Multinomial Naïve Bayes	Topic Modelling
Stacked Bar	56.10%	76.19%
Data Table	29.26%	7.14%
Pie Chart	14.63%	14.7%

the data table over the pie chart, feeling that the data table was more informative, especially for the Multinomial Naïve Bayes classifier.

A few of the participants provided answers to the question regarding if they felt that a visualization was missing something. One participant suggested that instead of showing percentage values, show the actual values as was done in the bar chart. Another participant commented that they found it to be too much work to click on the pie chart every time they wanted to see the detailed explanation for the word-based recommendations. A few participants commented that they preferred the word-based recommender to the topic-based recommender.

When examining the responses regarding trust across an individual user's session, we observed that for the first few times that they were presented with recommendations, their level of trust was low (e.g. ratings of 2). However, as they used the plug-in more, their level of trust increased (e.g. ratings of 5) quickly.

The results from our post-usage survey showed that more than half of the participants wanted to see more than one visual representation of the recommendations. Also, the majority (75%) felt that it was "very important" or "extremely important" to represent recommendations with explanations in a visual manner.[8]

5. Threats to Validity

Although in our study we trained our classifier using data from a single Mozilla project - `Bugzilla`, we do not feel that this limits the generalizability of our results. As our focus was on the representation of the recommendations, not the accuracy of the recommendations, our results are not dependent on the project used. Similarly, the study participants were from a wide range of occupational backgrounds, which further supports the generalizability of the results. Finally, generalizability related to using an open-source project vs. a commercial project or few projects vs. many projects were not deemed to be a concern.

That the participants in the user study were not associated with the Mozilla projects may have resulted in inaccurate feedback. As this was a pilot study, we plan to address this threat in a future study where we recruit project members of the dataset used for training the recommender system. Such a study is expected to provide more detailed comments regarding trust in the recommendations and if the information provided for explanation is sufficient for the task of bug report assignment.

There is a possibility that our results may suffer from social desirability bias (i.e. "please the researcher" bias). Based on the trend where participants initially reported that they had low trust in the recommendations and then the trust level improved, we do not feel that such bias had a significant impact overall. However, we cannot discount this possibility.

6. Conclusion

This work investigated the use of visualization for explaining bug report assignment recommenders. To accomplish this, we created a web service that provides explanations of assignment recommendations for two types of recommenders using three visualizations. We found that developers did prefer visual explanations, with 75% of participants stating that the visual explanations increased their understanding of the assignment recommendations. We also found that developers gained trust in the recommendations over time and that the developers preferred a stacked bar chart.

[8] See [23] for a more detailed discussion of the study results.

References

[1] S. Bostandjiev, J. O'"Tasteweights: a visual interactive hybrid recommender system," in *Proc. of the 6th ACM Conf. on Recommender Systems*, 2012, pp. 35–42.

[2] J. O'Donovan, B. Smyth, B. Gretarsson, S. Bostandjiev, and T. Höllerer, "Peerchooser: Visual interactive recommendation," in *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, 2008, pp. 1085–1088.

[3] K. Verbert, D. Parra, P. Brusilovsky, and E. Duval, "Visualizing recommendations to support exploration, transparency and controllability," in *Proc. of the 2013 Int'l Conf on Intelligent User Interfaces*, 2013, pp. 351–362.

[4] J. Xie, Q. Zheng, M. Zhou, and A. Mockus, "Product assignment recommender," in *Proc. of the 36th Int'l Conf. on Softw. Eng.*, 2014, pp. 556–559.

[5] J. L. Herlocker, J. A. Konstan, and J. Riedl, "Explaining collaborative filtering recommendations," in *Proc. of the 2000 ACM Conf. on Computer Supported Cooperative Work*, 2000, pp. 241–250.

[6] D. Parra, "Beyond lists: Studying the effect of different recommendation visualizations," in *Proc. of the 6th ACM Conf. on Recommender Systems*, 2012, pp. 333–336.

[7] C. T. Wolf and K. E. Ringland, "Designing accessible, explainable ai (xai) experiences," 2020.

[8] R. Hughes, C. Edmond, L. Wells, M. Glencross, L. Zhu, and T. Bednarz, "Explainable ai (xai): An introduction to the xai landscape with practical examples," in *SIGGRAPH Asia 2020 Courses*, 2020.

[9] N. Tintarev and J. Masthoff, "Effective explanations of recommendations: user-centered design," in *Proc. of the 2007 ACM Conf. on Recommender systems*, 2007, pp. 153–156.

[10] J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?" in *Proc. of the 28th Int'l Conf. on Softw. Eng.*, 2006, pp. 361–370.

[11] J. Anvik and G. C. Murphy, "Determining implementation expertise from bug reports," in *Proc. of the 4th Int'l Workshop on MSR*, 2007, pp. 2–10.

[12] C. C. Williams and J. K. Hollingsworth, "Bug driven bug finders," in *Proc. of the Int'l Workshop on MSR*, 2004, pp. 70–74.

[13] T. T. Nguyen, A. T. Nguyen, and T. N. Nguyen, "Topic-based, time-aware bug assignment," *ACM SIGSOFT Softw. Eng. Notes*, vol. 39, no. 1, pp. 1–4, 2014.

[14] C. Teyton, M. Palyart, J.-R. Falleri, F. Morandat, and X. Blanc, "Automatic extraction of developer expertise," in *Proc. of the 18th Int'l Conf. on Evaluation and Assessment in Softw. Eng.*, 2014, p. 8.

[15] C. Zhang, J. Yang, Y. Zhang, J. Fan, X. Zhang, J. Zhao, and P. Ou, "Automatic parameter recommendation for practical api usage," in *Proc. of the 34th Int'l Conf. on Softw. Eng.*, 2012, pp. 826–836.

[16] J. Anvik and G. C. Murphy, "Reducing the effort of bug report triage: Recommenders for development-oriented decisions," 2011.

[17] K. Oku, S. Nakajima, J. Miyazaki, and S. Uemura, "Context-aware svm for context-dependent information recommendation," in *Proc. of the 7th Int'l Conf. on Mobile Data Management*, 2006, p. 109.

[18] J. A. Xu and K. Araki, "A svm-based personal recommendation system for tv programs," in *12th Int'l Multi-Media Modelling Conf.*, 2006, pp. 4–pp.

[19] H. Zhang, F. Min, and S. Wang, "A random forest approach to model-based recommendation," *Journal of Information & Computational Science*, vol. 11, no. 15, pp. 5341–5348, 2014.

[20] H.-R. Zhang and F. Min, "Three-way recommender systems based on random forests," *Knowledge-Based Systems*, vol. 91, pp. 275–286, 2016.

[21] Z. Lu, D. Szafron, R. Greiner, P. Lu, D. S. Wishart, B. Poulin, J. Anvik, C. Macdonell, and R. Eisner, "Predicting subcellular localization of proteins using machine-learned classifiers," *Bioinformatics*, vol. 20, no. 4, pp. 547–556, 2004.

[22] C. Treude, P. Gorman, L. Grammel, and M.-A. Storey, "Workitemexplorer: Visualizing software development tasks using an interactive exploration environment," in *Proc. of the 34th Int'l Conf. on Softw. Eng.*, 2012, p. 1399–1402.

[23] S. A. Bhuyan, "Visual representation of bug report assignment recommendations," Master's thesis, University of Lethbridge, Lethbridge, Alberta, CANADA, 2019. [Online]. Available: https://hdl.handle.net/10133/5651