Replication: Do Developers Discuss Design?

Alvi Mahadi*, Neil Ernst*
*Department of Computer Science, University of Victoria, Victoria, Canada
{amahadi, nernst}@uvic.ca

***Abstract*—In this work, we aim to replicate the findings of [1]. There are mainly two steps of this replication. The first step is to build the classifier pipelines using Naive Bayes and Decision Tree and achieve the result that is reported in the original paper. After doing that, we show that the Decision Tree works better in terms of accuracy. Then we have conducted an empirical study on the data classified with the classifier to replicate the (a) proportion of design discussions, (b) criteria of developers who discuss design. We have discussed our findings while comparing them to the original work. We can see an almost similar result with [1].**

***Index Terms*—design, discussions, classification, naive bayes, decision tree, empirical study**

I. Introduction

Design is often mentioned to be very important to achieve certain properties and characteristics of a software system. Although it is very important to have design documentation, developers usually do not produce design documentation that often. But they discuss about different aspects of design in various communication channel like Github pull requests, issue tracker or commits. The paper we are trying to replicate provides quantitative evidence that developers address design through discussions. The authors of the original paper have mainly tried to answer two research questions:

1) To what extent do developers discuss design in open-source projects?
2) Which developers discuss design?

We are trying to strictly replicate the steps to answer those two research questions. In the process of answering the two main research questions, we are also answering some auxiliary research questions such as: **(a)** the ratio between design discussions vs contributors, **(b)** the proportion of all design discussions in a project to which a developer has contributed which is named as *Coverage* and **(c)** the correlation between the coverage and the commit for a particular developer.

We have organized the paper in the following manner. Section II lists some closely related works that are already done. Section III describe our dataset on which we are conducting our replication. Section IV explains our replication steps and transitions of the data. Section V briefly discusses the results and the similarity of our result with the original paper. Lastly, Section VI gives a summery of our paper and a concluding remark on the original paper that we have replicated.

II. Related Work

To the best of our knowledge, this is the first replication work that takes the identical approach to replicate the work in [1]. However, this study has been revisited once before by [2] to determine if it is possible to predict a single commit message is about design or not. The authors of [2] manually annotated their own dataset that they used as train data and implemented a variety of classifiers to determine the best performing classifier. As they did not used the same dataset and procedures as [1], they did not get the accuracy which was 87.58% for their best performing classifier, Random Forest.

If we do not limit ourselves to strict replication, we can find some literature which implements similar work. Maldonado in [3] tried to identify self admitted technical debt which can be perceived as one kind of design issue using Natural Language Processing. Like [1], they manually classified a set of data which is used as training data for the Stanford Classifier, which is a Java implementation of a maximum entropy classifier. Using this classifier they achieved 90% accuracy on test data and found out that, 23% of the comments in both design and requirement comments are technical debt. A recent manual investigation on Github discussion by Viviani [4] shows some detailed information about the design discussions. He conducted an empirical study on the discussions and matched their result with [1] to validate their claim of 23-24% design discussions being present in the discussions.

III. Data Set

The authors of the original paper took 90 projects that was present in the GHTorrent data set [5] and discarded 13 projects due to having less than 50 discussions. In total they selected 77 projects with 102,122 discussions. Then out of the 77 projects, they randomly selected 5 projects. Then they randomly selected 200 discussions from each project, totaling 1,000 discussions for manual classification. Then the 1,000 discussions were manually classified by two individual authors of the paper. After the manual classification, 967 discussions were taken in which the classification by both authors matched. The 1000 discussions they took were randomly taken from five projects. Also they did not discuss or specify any rules for the manual classification of the 1,000 discussions. Due to the randomness of the data and the classification choice being very subjective, it is impossible to replicate the data they used. So we decided to contact them for their corpus of 1000 discussions and they provided us with the data. Also the 77 projects they took were not specified either. So we used their processed data from the GHTorrent database.

discussions.csv— contains the 967 sentences that are manually classified by two of the authors of the paper, 226 (23%) of these refer to 'design' points and 741 (77%) of them are

other issues labeled as 'general'. A sample of the file is shown in Table I

rq_1.csv— is a data file with 5,86,839 lines of data. Each line represents three columns separated by a space. The first column points to the repository and project name separated by '/'. The second column provides information about whether this discussion comes from an issue, commit or pull request with the associated commit, issue or pull request number. The third column represents the label that the classifier assigned. Table II shows a sample of the data that this file contains.

rq_2_a.data— also contains 5,86,839 lines of data divided into four columns by single space that represents username of the contributor, commit/issue/pull request with respective number, label of the discussion and repository name/project name respectively as shown by the sample in Table III.

rq_2_b.data— is a narrowed copy of 'rq_2_a.data' that contains the username of the contributor, repository/project name and commit, issue or pull_request with number that points to only the design label.

rq_2_a_c.data— hold 23,293 lines of data that is divided into three columns. The first column represents the repository name/project name, the second column shows the user name of the contributor. The last column points to the number of design discussions the user in the previous column addressed in the project defined by the first column. Table IV shows the sample of data in the file.

IV. Replication

We strictly followed the steps demonstrated in the paper. While replicating, we took the following steps:

A. Stopwords Removal

Stopwords are words that have the similar likelihood of occurring in documents regardless of the relevance of the query [6]. It is very important to remove stopwords to improve the performance of the classifier as well as reducing the size of the dataset. We have used the The *Natural Language Toolkit's (NLTK)*[1] [7] english stopwords set to primarily remove some general stopwords. Then we have looked for some document specific stopwords ex. 'lgtm' which is short form of 'looks good to me' and removed them. The before and after status of a sentence for stopwords removal can be seen in Table V

B. Feature Selection

Feature selection is the process of selecting a subset of the terms occurring in the training set and using only this subset as features in text classification[2]. Feature selection are used mainly for twos reason. **One**, it reduces the size of the effective vocabulary which is helpful to speed up the training for an expensive classifier like Decision Tree. **Two**, it often increases the accuracy by reducing the amount of noisy feature in a corpus. In the original paper, they combined two methods of feature extraction. **First**, they have found and ranked the bigram[3] collection of other association measures by first constructing it for all bigrams in a given sequence. Then they have provided the Pearson's chi-square as the score to return the top bigrams. **Second**, they have counted every bigram by iterating through every word in the sentence to find out the pair and assigned a true/false value for them. Table VI shows the dictionary after implementing combined bigram features.

C. Classification and Validation

We have used Naive Bayes and Decision Tree classifier as the classification models for the sentences like they did in the paper. Also following them, we have also used 10-fold cross validation method to validate the performance of the classifiers. We have divided the whole dataset in 10 chunks containing almost 100 sentences in each chunk. Then we have taken 9 chunks to train our model and then tested it with the other chunk we left untouched. We repeated this procedure for 9 more times each time moving onto the next chunk to use it as test and the rest of them as train. We are calculating the accuracy of every pass which is essentially the fraction of corresponding values that are equal to the test label. Finally, we have calculated the mean of 10 passes to estimate the overall accuracy for each classifier. We have also estimated the performance of each classifier in terms of time. To do that, we have taken into account the difference between the start time of the classification and the end time. This can demonstrate the cost of each classifier in time.

D. Information Extraction

Due to the lack of clarification in the paper about which 77 projects were used and due to the randomness in choice of the 1,02,122 discussions that were taken from the 77 projects, we used the pre-processed dataset from [1] to extract the information needed for answering the research questions. We used a quantitative approach to extract different information from the *.data* files.

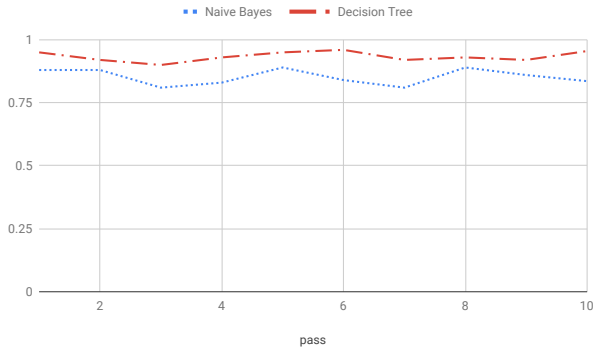


Fig. 1: Accuracy of The 10-fold Validation

[1]NLTK, https://www.nltk.org/

[2]https://nlp.stanford.edu/IR-book/html/htmledition/feature-selection-1.html

[3]https://en.wikipedia.org/wiki/Bigram

TABLE I: Data sample in discussions.csv

Discussion Text	Label
Did you send a bug report comment upstream or something because otherwise we are gonna have to fix this again with the next version of assimp SSL contexts shouldnt be reused across connections see So its probably more appropriate to pass in factories directly	general design

TABLE II: Data sample in rq_1.data

Repository/Project	Event Category with Number	Label
akka/akka	akka/akka-commit_comments-d9e0088f3cc537ea342f6fc4e99ee5319dfc94ae	general
akka/akka	akka/akka-commit_comments-3ce3f270dfce5da7aa5b6270b0559e2c3c0fff6f	design
ariya/phantomjs	ariya/phantomjs-issue_comments-10045	design
antirez/redis	antirez/redis-issue_comments-160	general

TABLE III: Data sample in rq_2_a.data

Username	Event Category with Number	Label	Repository/Project
viktorklang	akka/akka-commit_comments-d9e0088f3cc537ea342f6fc4e99ee5319dfc94ae	general	akka/akka
jboner	akka/akka-commit_comments-a3026b3316dc5b34c3d37ce6fc56cc44bac1d561	design	akka/akka
antirez	antirez/redis-issue_comments-646	general	antirez/redis
patriknw	akka/akka-pull_request_comments-149	general	akka/akka

TABLE IV: Data sample in rq_2_a_c.data

Repository/Project	Username	Involvement in design discussions
zurb/foundation	stewarty	1
rails/rails	hassox	1
rails/rails	stjhimy	1
twitter/finagle	kolbasov	1
cakephp/cakephp	atkrad	1
zendframework/zf2	ravids	1
symfony/symfony	jdhoek	6
joyent/node	paulfryzel	1
mxcl/homebrew	hackdefendr	9

TABLE V: Sample of sentence status before and after stopwords removal. The sentence on the top is the actual sentence. The bottom one is after removing the stopwords

Did you send a bug report comment upstream or something because otherwise we are gonna have to fix this again with the next version of assimp
send bug report comment upstream something otherwise gonna fix next version assimp

TABLE VI: Sample of the dictionary of words that is returned after using combined bigram features. In the top: the original text after stopwords removal. In the bottom: The dictionary of combined word bigrams

martijnvg seems reasonable done
‘martijnvg’: True, ‘seems’: True, ‘reasonable’: True, ‘done’: True, (‘martijnvg’, ‘seems’): True, (‘reasonable’, ‘done’): True, (‘seems’, ‘reasonable’): True

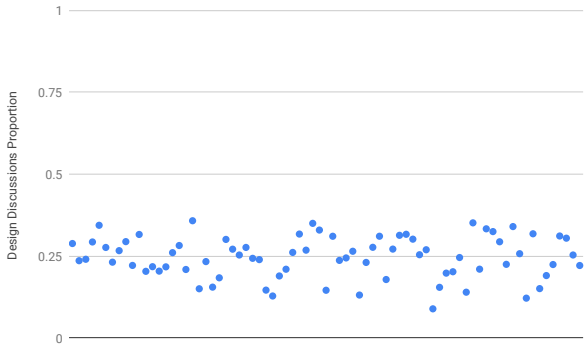
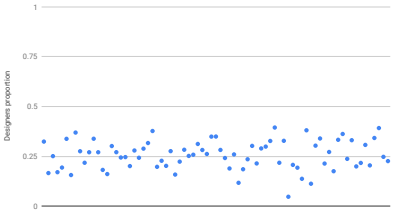


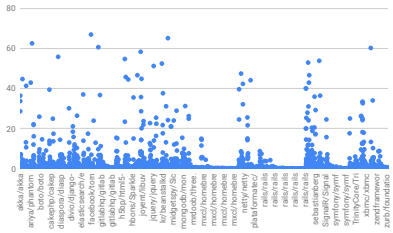
Fig. 2: Proportion of Design Discussions Per Project

V. Results

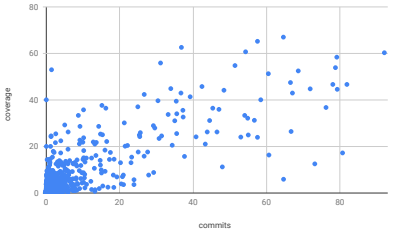
After executing the 10-fold cross validation we can see that Decision Tree performs better than Naive Bayes and our result matches the result in [1]. We achieved 94 $\pm$ 1% for Decision Tree and 86 $\pm$ 1% for the Naive Bayes classifier which matches with the original paper. The accuracy of each fold of the two classifier can be visualized in Fig. 1 which clearly shows that Decision Tree outperforms Naive Bayes in every pass. Also as per performance in terms of time, Naive Bayes outperformed Decision Tree by completing almost all the experiments in less than 1 second where Decision Tree took as long as almost 10 minutes in some cases in the same environment.



(a) Proportion of Design Discussions Contributors



(b) Developer's Design Discussions Coverage. X-Axis Denotes the Projects and Y-Axis Refers to the Percentage of Coverage



(c) Developers Coverage with Respect to Commits

Fig. 3: Empirical Results

A. RQ1: To what extent do developers discuss design?

We have used their processed data to determine the proportion of design discussions for every project. Fig. 2 shows the proportion of design discussion we can see that, overall $25 \pm 6\%$ of the discussions in a project points to some kind of design aspect. We have also observed that our result completely matches with [1].

B. RQ2: Which developers discuss design?

The authors of [1] analyzed data related to 22,789 developers from the 77 projects that they used. There is no information of how the developers were chosen. So we could not replicate their database pre-processing stage. So, we used their processed data to conduct the empirical study similar to them. After conducting the study, we found very similar results found by them. Similarly, we found that the number of developers involved in design discussions is 8,207 (36%). Fig. 3 (a) shows the proportion of contributors for each project that got involved in design discussions.

In the later experiments, they introduced a term called *Coverage* which indicates the proportion of all design discussions in a project to which a developer has contributed. For example, if a project has 10 design discussions and a developer contributes to 5 of them then the coverage of this developer would be 0.5. In the second step, they found that the majority of developers contributes to less than 10% of the design discussions. They also found that 99% of the developers contribute to only 15% of the design discussions in a specific project. After replicating this, we also have the similar conclusion. Fig. 3(b) shows the coverage of developers for a specific project.

The authors in [1] hypothesized that several factors could be present for developers to contribute to a broad range of design discussions. This scenario implies that these developers have a key role in the projects. They measured the relationship between the proportion of developers commits and their respective coverage. They used Spearman's method which we also used. Eventually we found the same result as they did regarding the relation of developers commits and their coverage. Fig 3(c) plots the relation between the percentage of developer commits and their coverage. We found 74% correlation between those two parameters which is exactly the same as theirs.

VI. Conclusions

In this work, we tried to replicate the work in [1] as strictly and identically as possible. In this process, **first** we have replicated their steps to build the classifier they used. After building the classifier, we observed that the accuracy they reported exactly matches with the accuracy we obtained. Then in the **second** phase, we tried to replicate the empirical study conducted with the classified document. After replicating the study we verified all the results they mentioned in their paper. After doing this study we have come to the conclusion that this study is highly replicable.

Acknowledgment

This work was conducted to fulfill the requirements of graduate assignment of CSC 578A under Professor Neil Ernst. We would like to thank the authors of [1] in particular João Brunet, for providing us with their dataset.

References

[1] J. a. Brunet, G. C. Murphy, R. Terra, J. Figueiredo, and D. Serey, "Do developers discuss design?" in *Proceedings of the 11th Working Conference on Mining Software Repositories*, ser. MSR 2014. New York, NY, USA: ACM, 2014, pp. 340–343.

[2] A. Shakiba, R. Green, and R. Dyer, "Fourd: Do developers discuss design? revisited," in *Proceedings of the 2Nd International Workshop on Software Analytics*, ser. SWAN 2016. New York, NY, USA: ACM, 2016, pp. 43–46.

[3] E. d. S. Maldonado, E. Shihab, and N. Tsantalis, "Using natural language processing to automatically detect self-admitted technical debt," *IEEE Transactions on Software Engineering*, vol. 43, no. 11, pp. 1044–1062, Nov 2017.

[4] G. Viviani, C. Janik-Jones, M. Famelis, X. Xia, and G. C. Murphy, "What design topics do developers discuss?" in *Proceedings of the 26th Conference on Program Comprehension*, ser. ICPC '18. New York, NY, USA: ACM, 2018, pp. 328–331.

[5] G. Gousios, "The ghtorent dataset and tool suite," in *Proceedings of the 10th Working Conference on Mining Software Repositories*, ser. MSR '13. Piscataway, NJ, USA: IEEE Press, 2013, pp. 233–236.

[6] W. J. Wilbur and K. Sirotkin, "The automatic identification of stop words," *Journal of Information Science*, vol. 18, no. 1, pp. 45–55, 1992.

[7] E. Loper and S. Bird, "Nltk: The natural language toolkit," in *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ser. ETMTNLP '02. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp. 63–70.