# Framework for Automatic Bug Classification in Bug Triage System

[1]nikita R. Hiwse, [2] Prof. Roshani Talmale
[1]wireless Communication And Computing,[2]computer Science & Engineering
Tulsiramji Gaikwad-Patil College Of Engineering & Technology, Nagpur
*Email: [1]hiwsenikita04@Gmail.Com,[2] Hod.Cse@Tgpcet.Com*

***Abstract:*** A Software bug is an error, flaw, failure or fault in a computer program or system that causes it to produce an incorrect or unexpected result. When bugs arise, we have to fix them which is not easy. Most of the companies spend 40% of cost to fixing bugs. The process of fixing bug is bug triage or bug assortment. Triagingthis incoming report manually is error prone and time consuming .Software companies spend most of their cost in dealing with these bugs. In this paper we classifying the bugs so that we can determine the class of the bug at which class that bug is belongs and after applying the classification we can assign the particular bug to the exact developer for fixing them. This is efficient. In this paper we are using combination of two classification techniques ,naïve Bayes (NB) and k nearest neighbor(KNN).In modern days company uses automatic bug triaging system but in Traditional manual Triaging system is used which is not efficient and taking too much time .For triaging the bug we require bug detail which is called bug repository. In this paper we also reducing the bug dataset because if we having more data with unused information which causes problem to assigning bugs. For implementing this we use instance selection and feature selection for reducing bug data. This paper describe the whole procedure of bug allotment from starting to end and at last result will show on the basis of graph .Graph represents the maximum possibility of class means at which class the bug will belongs.

***Keywords-****bug triage, bug data reduction, bugs classification technique (NB &KNN).*

*****

## INTRODUCTION-

**E**veryday new bugs are generated and fixing these bugs are more difficult.Software companies are for developing new applications but here they waste their time for fixing the bugs.In this paper we triaging the bug by using some classification techniques and we also reducingthe bug dataset.For triaging the bug we require the bug details so that we can assign the bug to the particular developer which is more efficient. Bug details are stored in bug repository.In traditional days bugs are assign manually which is not efficient and also time consuming.they could not handle the large dataset .In modern days bug assignment is done automatic but still there was a problem, details of the bugs are not properly .Some of the information is incorrect and some are unused .we could not predict the class of the bug depending on that information so in this paper we find the solution of that problem and the solution is "reduction of bug dataset". Where reducing unwanted data which is not useful for triaging the bug. This process is comes under the prepossession.

Applying the example choice procedure to the information set can diminish bug reports yet the accuracy of bug triage might be diminished; applying the element determination system can lessen words in the bug information and the exactness can be expanded.In the meantime, consolidating both strategies can build the accuracy, and additionally decrease bug reports and words. For instance, when 50 percent of bugs and 70 percent of words are evacuated, the exactness of Naive Bayes on Eclipse enhances by 2 to 12 percent and the accuracy on Mozilla enhances by 1 to 6 percent. In view of the featuresfrom historical bug information sets, our prescient model can give the exactness of 71.8 percent for foreseeing the lessening request. In view of top hub investigation of the properties, results demonstrate that no individual characteristic can decide the decrease request.

Basically bug repository is database where we store information regarding bugs.Differentbugs having different information so detecting a class of that bug we require bug repository or bug database. In a bug repository, a bug is maintained as a bug report, which records the textualdescription of reproducing the bug and updates according to the status of bug fixing [20]. A bug repository consist of many types of tasks on bugs, e.g. fault prediction [7], [18], bug localization [3], and reopenedbuganalysis [19]. Bug reports in a bug repositoryare called bug data.

This classification process is not done by a single person, for making it successful it required Admin, Developer and Tester. Tester is responsible for testing the program whenever bug is occur, tester submit the bug with its bug details in the bug repository or bug database. By using this information about bug, classification process is done for determine the class of that bugand after that administrator assign the bug depending on the class to the proper developer.

Important stages in the bug triaging process is selecting the most appropriate developer to fix a new bug report and it has a significant effect in decreasing the time taken for the bug fixing process [16] and the cost of the projects [2], [11].Software companies spend over 45 percent of cost in fixing bugs [1], [8], and [17]. In traditional bug triage systems, a developer whois dominant in all parts of the project as well as the activities plays the role of bug triager in the project. The triager reads a new bugreport, makes a decision about the bug, and then selects the most appropriate developer who can resolve the bug. Fixing bug reports through the traditional bug triage system is very time consuming and also imposes additional cost on the project [2].

One of the most important reasons why bug triaging is such a lengthy process is the difficulty in selection of the most competent developer for the bug kind. The bug triager, the person who assigns the bug to a developer, must be aware of the activities (or interest areas) of all the developers in the project. Bug triaging normally takes 8 weeks to resolve a bug if the developer, to whom the bugreport is assigned,

could not resolve it, it is assigned to another developer. This would consume both time and money. Thus, it is really important on part of bug triager to assign the bug report to a developer who could successfully fix the bug without need of any tossing. Hence, the job of bug triager is really crucial [2].

In this paper we using the classification to assigning the bug which is automatic process and by using this bug fixing is done quickly as compare to the manualassignment.

Depending on graph we can assign the bug easily, here two graphs are use first graph is represent maximum possibility of class based on word count (KNN) and second graphs describe the combination of KNN & NB which is based on frequency. The output of second graph is generated from the help of first graph .Final output will beconsider from second graph.

## METHODS AND MATERIAL

### NB Classifier

Naïve bayes classifier is developed by "Thomas bayes" which is use for classification process.Bayes classifier is also called as Idiot Bayes, Naïve Bayes and Simple Bayes. Naïve Bayes is fast,space efficient and easy to learn. Assume that we have two classes c1 = male, and c2 = female. We have a person whose sex we do not know, say "sonu" . Classifying sonu as male or female is equivalent to asking is it more probable that sonu is male or female, i.e which is greater p(male| sonu) or p(female| sonu).To solve this Problem we will use dataset (Table a)where data about sex is present .classifier calculate the maximum count with the use of dataset and declare the output. Here SONU is female because in table here is 2 entries of sonu with sex "female" and only one entry for male.

| Name | Sex |
|---|---|
| Sonu | Female |
| Monu | Male |
| Sonu | Female |
| Neha | Female |
| Sonu | Male |

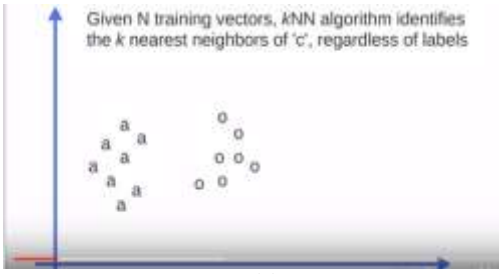
Table(a)

Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlation between the color, roundness and diameter features.

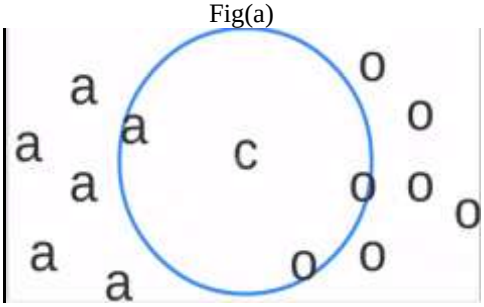
Similarly this classifier is helpful in classifying the flaw or bug and with the help of dataset where all type of classes is inserted it predict the class.An advantage of naive Bayes is that it only requires a small amount of training data to estimate the parameters necessary for classification.

### KNN Classifier

KNN Means K nearest neighbor which identify the k nearest neighbor of 'c' where c is the item or bug (flaw).With the help of this classifier we classify the bug .For ex: the value of k is 3,here are two classes a and o and we have to find out the class for c.here k=3 means we need to find out 3 nearest neighbor of c.




Fig(a)




fig(b)

In fig(a) class a &b is present and In fig(b) c is present which is one type of bug and after applying the KNN it defines 3 nearest neighbor ,one from class

a and two from class o. Here 2 vote for class o and 1 vote for class a so output will be class o means bug c is belongs to class o by using KNN.

In this paper we are using combination of these two algorithm for classification where KNN algorithm is based on word count and NB algorithm is based on term frequency.

### Applying Instance Selection and FeatureSelection

In this paper we use the instance selection and feature selection for reducing bug data set. For triaging the we require the dataset and we need to convert it into text matrix with two dimensions,bug dimension and word dimension. In this we combine the instance and feature to generate the reduced data set. Instance selection and feature selection are widely usedtechniques in data processing for replacing original dataset with reduced data set for bug triage.instance selection is to obtain a subset ofrelevant instances (i.e., bug reports in bug data) [21] while feature selection aims to obtain a subset of relevant features (i.e., words in bug data) [22]. In our work, we employ the combination of instance selection and feature selection. Todistinguish the orders of applying instance selection and feature selection, we give the following denotation. Givenan instance selection algorithm IS and a feature selectionalgorithm FS, we use FS!IS to denote the bug data reduction,which first applies FS and then IS; on the other hand,IS!FS denotes first applying IS and then FS.

**Word dimension:** We use feature selection to remove noisy duplicate words in a data set. By removinguninformative words, feature selection improves the accuracy of bug

triage. This can recover the accuracy lossby instance selection.

**Bug dimension:** Instance selection can remove uninformative bug reports; meanwhile, we can observe that the accuracy may be decreased by removing bug reports.

**Algorthm:**

**Algorithm Preprocess (Data D)**

Step 1: Read Data into Array
Step 2: Remove All Stop words
$\Sigma\, i = 0 \mid \phi\, n \neq stop(i)$
Step 3: Remove Redundancy from Array
$\Sigma\, i = 0 \mid \phi\, n \neq repeat(i)$
Step 4: Remove all Special Symbol and digits.
Step 5: Write back

**Algorithm Classification (Data D)**

Step 1: Read Data into Array
Step 2: Call Preprocess (D)
Step 3: Calculate Word count
$\Sigma\, i = 0 \mid \phi\, i = i + 1\; if\; a[i] \in final[i]$
Step 4: Calculate Frequency
Tf = number of occurrences / total words
Step 5: Calculate Normalized TF
NTF = sum of Tf / number of classes
Step 6: Generate Decision Matrix
Step 7: Calculate Final max class value and classify.

**Module**

**Module1: Dataset Generation**- First module of this project is dataset generation. all types of classes are inserted in this dataset.we introduce 6 types of classes. Bad practice, Correctness, dodgy code, Experimental, Internationalization, and Malicious code Vulnerability.
If bug belongs to this classes it automatically assign to that class of developer which is efficient for fixing the bug.

**Module2: Prepossessing** second module of this project is prepossessing. In this module we remove all stop words which is not useful for detecting the class.
After prepossessing we will apply the classifier on that preprocessed data for detecting the class.

**Module3: Bug Classification:** This is third module "bug Classification". Two classification algorithms are use for classifying the bug.NB(naïve bayes) and KNN(K nearest neighbor).In this project we are using the Combination of these two algorithm due tothis, it generate the better result.

**Module 4: Bug allotment**: last module is Bug allotment where we are allotting the bug to the particular developer depending upon their class.

**Overview of Dataset**

We have used FINDBUGS Categories as our bug dataset for unstructured bug categories.
Some of Bug categories

**Correctness bug**

Probable bug - an apparent coding mistake resulting in code that was probably not what the developer intended. We strive for a low false positive rate.

**Bad Practice**

Violations of recommended and essential coding practice. Examples include hash code and equals problems, cloneable idiom, dropped exceptions, serializable problems, and misuse of finalize. We strive to make this analysis accurate, although some groups may not care about some of the bad practices.

**Dodgy**

Code that is confusing, anomalous, or written in a way that leads itself to errors. Examples include dead local stores, switch fall through, unconfirmed casts, and redundant null check of value known to be null. More false positives accepted. In previous versions of FindBugs, this category was known as Style.

**Probable bug** - an apparent coding mistake resulting in code that was probably not what the developer intended. We strive for a low false positive rate.

**Experimental Results**

• We introduce the issue of information lessening for bug triage. This issue means to increase the information set of bug triage in two viewpoints, to be specific
a) To at the same time lessen the sizes of the bug measurement and the word measurement.
b) to enhance the precision of bug triage.
• We propose a mix way to deal with tending to the issue of information diminishment. This can be seen as a use of example choice and highlight determination in bug vaults.
• We manufacture a parallel classifier to foresee the request of applying case choice and highlight choice. As far as anyone is concerned, the request of applying occurrence determination and highlight choice has not been explored in related spaces.
By utilizing the programmed bug triage approach the time and cost is spared. This framework is executed by utilizing the Naive bayes grouping strategy. The task of bug to designer is done naturally. Nature of bug triage is enhanced utilizing this new framework, in light of the fact that here we utilize classifier for execution. We are utilizing occasion and highlight determination for information decrease. Occurrence determination and Feature choice with Naïve Bayes and KNN grouping for bug triage.
The proposed system is implemented in Java and MySQL. The dataset is provided with 9 classes from FindBugs 2.0 that is openly available on GitHub and Source Forge. The algorithm used for classification is KNN and Naïve Bayes and we also provide comparative study for both this algorithms in terms of Bug Triage System.

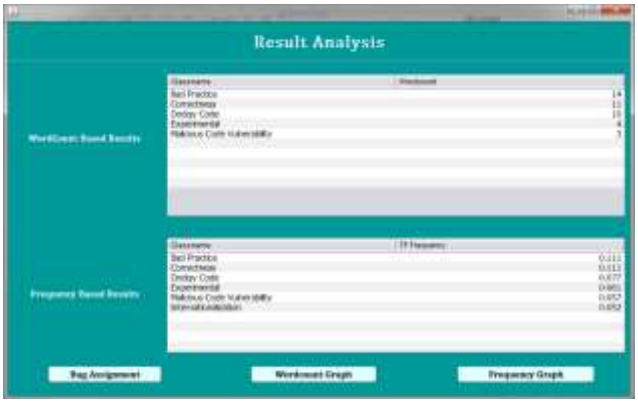
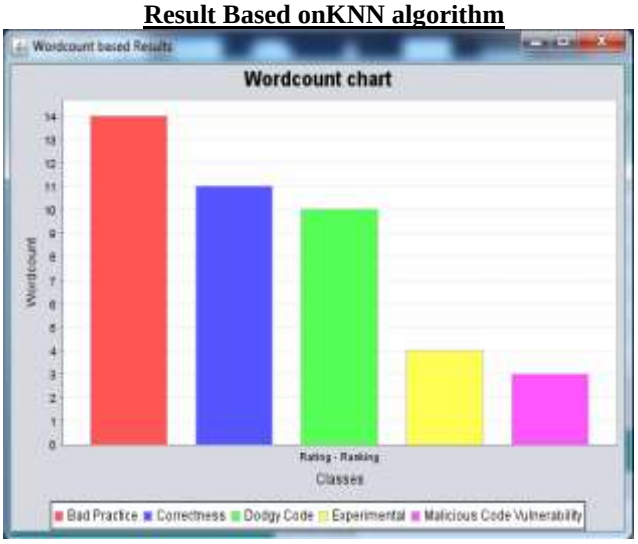
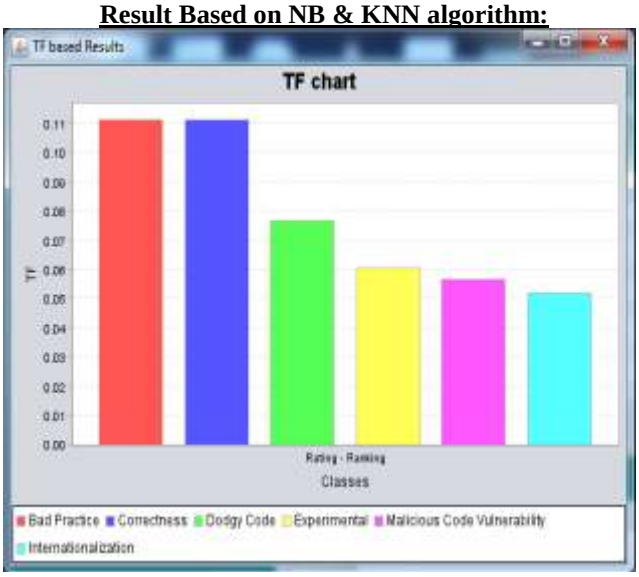



Fig: KNN and NB Results

**Result Based onKNN algorithm**




**Result Based on NB & KNN algorithm:**




Here two Graphs are Present 1st one is based on word count which support KNN Classifier and 2nd is based on Term Frequency which support NB & KNN classifier .the result is "bad Practice" means the bug is belongs to the class which name is "bad Practice".


Bug allotment

Bug allotment is last step where we allot bugs to the developer for fix them.

**Conclusion and Future Scope:**

In this paper we introduced the concept of bug classification for allotting the bug to appropriate developer for fixing it and for this purpose we used the combination of two algorithms NB and KNN. We also introduced the reduction process of bug database by applying prepossessing.

Bug tracking systems are an important part of how teams in open source interact with their user communities. This interaction goes beyond users simply submitting bugs. Many follow-up questions are posed to the reporters of bugs andOften, if a reporter does not play an active role in the discussion of the bug, little progress is made. Our results highlight the importance of effectively and efficiently engaging the user community in bug fixing activities, and keeping them up-to-date about the status of a bug. We believe that our results will help to form the design of new bug tracking systems that will aim at eliciting the right information fromUsers and facilitating communication between end users and developers as well as among developers. An integration and active participation of users in bug tracking will result in bugs being fixed faster and more efficiently.

Bug triage is a costly stride of programming upkeep in both work cost and time cost. In this paper, we join highlight choice with occasion determination to diminish the size of bug information sets and enhance the information quality. To decide the request of applying example determination and highlight choice for another bug information set, we separate traits of every bug information set and prepare a prescient model in view of verifiable information sets. We experimentally examine the information lessening for bug triage in bug storehouses of FindBugs Database of Bugs. Our work gives a way to deal with utilizing strategies on information handling to shape diminished and brilliant bug information in programming advancement and upkeep. We also provide a system that can classify bugs with the help of KNN and NB classifier system. Bug Triage with automated classification is the main objective of proposed system.

## References


[1] Jifeng Xuan, He Jiang, Yan Hu, Zhilei Ren, Weiqin Zou, Zhongxuan Luo, and Xindong Wu, ―Towards Effective Bug Triage with Software Data Reduction Techniques,‖ IEEE Transactions, Volume 27, NO. 1, JANUARY 2015.

[2] Anjali, Sandeep Kumar Singh, ―Bug Triaging: Profile Oriented Developer Recommendation‖, International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163, Volume 2, Issue 1, January 2015.

[3] S. Artzi, A. Kie_zun, J. Dolby, F. Tip, D. Dig, A. Paradkar, and M. D.Ernst, "Finding bugs in web applications using dynamic test generation and explicit-state model checking," IEEE Softw., vol. 36, no. 4, pp. 474–494, Jul./Aug. 2010.

[4] Pankaj Rana, Asst. Prof. Saurabh Sharma "Review of Bug Serverity Prediction Techniques Using Data Mining"Volume 5, Issue 6,June2015 .

[5] J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?" in Proc. 28th Int. Conf. Softw. Eng., May 2006, pp. 361–370.

[6] J. Anvik and G. C. Murphy, "Reducing the effort of bug report tri- age: Recommenders for development-oriented decisions," ACM Trans. Soft. Eng. Methodol., vol. 20, no. 3, article 10, Aug. 2011.

[7] P. S. Bishnu and V. Bhattacherjee, "Software fault prediction using quad tree-based k-means clustering algorithm," IEEE Trans. Knowl. Data Eng., vol. 24, no. 6, pp. 1146–1150, Jun. 2012.

[8] Pankaj Gakare, Yogita Dhole,Sara Anjum, ―Bug Triage with Bug Data Reduction‖, International Research Journal ofEngineering and Technology (IRJET) e-ISSN: 2395 -0056,Volume 02 Issue 04, July 2015.

[9] C. C. Aggarwal and P. Zhao, "Towards graphical models for text processing," Knowl. Inform. Syst., vol. 36, no. 1, pp. 1–21, 2013.

[10] D. Cubranic and G. C. Murphy, "Automatic bug triage using text categorization," in Proc. 16th Int. Conf. Softw. Eng. Knowl. Eng., Jun. 2004, pp. 92–97.

[11] J. Anvik and G. C. Murphy, ―Reducing the effort of bug report triage: Recommenders for development-oriented decisions,‖ ACM Trans. Softw. Eng. Methodol., vol. 20, no. 3, pp. 10:1–10:35, Aug. 2011.

[12] A. K. Farahat, A. Ghodsi, M. S. Kamel, "Efficient greedy feature selection for unsupervised learning," Knowl. Inform. Syst., vol. 35, no. 2, pp. 285–310, May 2013.

[13] K. Gao, T. M. Khoshgoftaar, and A. Napolitano, "Impact of data sampling on stability of feature selection for software measure- ment data," in Proc. 23rd IEEE Int. Conf. Tools Artif. Intell., Nov. 2011, pp. 1004–1011.

[14] J. A. Olvera-Lopez, J. A.Carrasco-Ochoa, J. F. Martınez-Trinidad, and J. Kittler, "A review of instance selection methods," Artif. Intell. Rev., vol. 34, no. 2, pp. 133–143, 2010.

[15] C. Sun, D. Lo, S. C. Khoo, and J. Jiang, "Towards more accurate retrieval of duplicate bug reports," in Proc. 26th IEEE/ACM Int. Conf. Automated Softw. Eng., 2011, pp. 253–262.

[16] G. Jeong S. Kim, and T. Zimmermann, ―Improving bug triage with bug tossing graphs‖, in Proceedings of Seventh joint meeting ofEuropean Software Engineering Conference & ACM SIGSOFT symposium on Foundations of software engineering, ser. ESEC/FSE '09. New York, NY, USA: ACM, pp. 111–120, 2009.

[17] R. S. Pressman, ―Software Engineering: A Practitioner's Approach‖, 7th ed. New York, NY, USA: McGraw-Hill, 2010.

[18] S. Shivaji, E. J. Whitehead, Jr., R. Akella, and S. Kim, "Reducing features to improve code change based bug prediction," IEEE Trans. Soft. Eng., vol. 39, no. 4, pp. 552–569, Apr. 2013

[19] T. Zimmermann, N. Nagappan, P. J. Guo, and B. Murphy, "Characterizing and predicting which bugs get reopened," in Proc. 34th Int. Conf. Softw. Eng., Jun. 2012, pp. 1074–1083.

[20] T. Zimmermann, R. Premraj, N. Bettenburg, S. Just, A. Schr€oter, and C. Weiss, "What makes a good bug report?" IEEE Trans. Softw. Eng., vol. 36, no. 5, pp. 618–643, Oct. 2010.

[21] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for activelearning," Knowl. Inform. Syst., vol. 35, no. 2, pp. 249–283, 2013.

[22] I. Guyon and A. Elisseeff, "An introduction to variable and featureselection," J. Mach. Learn. Res., vol. 3, pp. 1157–1182, 2003.

[23] A. Srisawat, T. Phienthrakul, and B. Kijsirikul, "SV-kNNC: An algorithm for improving the efficiency of k-nearest neighbor," in Proc. 9th Pacific Rim Int. Conf. Artif. Intell., Aug. 2006, pp. 975–979.

[24] J. Tang, J. Zhang, R. Jin, Z. Yang, K. Cai, L. Zhang, and Z. Su, "Topic level expertise search over heterogeneous networks," Mach. Learn., vol. 82, no. 2, pp. 211–237, Feb. 2011.

[25] I. H. Witten, E. Frank, and M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, 3rd ed. Burlington, MA, USA: Morgan Kaufmann, 2011.

[26] D. R. Wilson and T. R. Mart_ınez, "Reduction techniques for instance-based learning algorithms," Mach. Learn., vol. 38,pp. 257–286, 2000.

[27] X. Wang, L. Zhang, T. Xie, J. Anvik, and J. Sun, "An approach to detecting duplicate bug reports using natural language and execution information," in Proc. 30th Int. Conf. Softw. Eng., May 2008,pp. 461–470.

[28] J. Xuan, H. Jiang, Z. Ren, and Z. Luo, "Solving the large scale next release problem with a backbone based multilevel algorithm," IEEE Trans. Softw. Eng., vol. 38, no. 5, pp. 1195–1212, Sept./Oct.2012.

[29] J. Xuan, H. Jiang, Z. Ren, J. Yan, and Z. Luo, "Automatic bug triageusing semi-supervised text classification," in Proc. 22nd Int.Conf. Softw. Eng. Knowl. Eng., Jul. 2010, pp. 209–214.

[30] J. Xuan, H. Jiang, Z. Ren, and W. Zou, "Developer prioritization inbug repositories," in Proc. 34th Int. Conf. Softw. Eng., 2012, pp. 25–35.