Categories of Automated Bug Report Management Techniques

Notes:

1) Total: 109 relevant papers published on 14 conferences and journals from 2006-2017:
   - 10 conferences: ICSE, FSE, ASE, ICSME, MSR, SANER(WCRE, CSMR-WCRE), ESEM, ICPC, ISSTA, ICST.
   - 4 journals: TOSEM, TSE, EMSE, and JSS.

   Filtering criteria:
   - Papers should be full research papers.
   - Papers should mainly focus on handling bug reports.
   - Papers should propose automated techniques (e.g., building prediction models or tools) which help to cope with bug reports.

2) All the 109 papers are research papers that propose automated bug report management techniques to help developers better manage bug reports.

3) 109 papers belong to 15 categories, of which 5 categories only has one paper.

1) Bug localization

(These techniques process a bug report, and locate relevant source code files or methods that possibly contain the bug.)

1. Moreno L, Treadway J J, Marcus A, et al. On the use of stack traces to improve text retrieval-based bug localization[C]//Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on. IEEE, 2014: 151-160.
2. Ye X, Bunescu R, Liu C. Mapping bug reports to relevant files: A ranking model, a fine-grained benchmark, and feature evaluation[J]. IEEE Transactions on Software Engineering, 2016, 42(4): 379-402.
3. Dilshener T, Wermelinger M, Yu Y. Locating bugs without looking back[C]//Proceedings of the 13th International Conference on Mining Software Repositories. ACM, 2016: 286-290.
4. Le T D B, Oentaryo R J, Lo D. Information retrieval and spectrum based bug localization: better together[C]//Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. ACM, 2015: 579-590.
5. Davies S, Roper M, Wood M. Using bug report similarity to enhance bug localisation[C]//Reverse Engineering (WCRE), 2012 19th Working Conference on. IEEE, 2012: 125-134.
6. Kim D, Tao Y, Kim S, et al. Where should we fix this bug? a two-phase recommendation model[J]. IEEE transactions on software Engineering, 2013, 39(11): 1597-1610.
7. Wang S, Lo D. Version history, similar report, and structure: Putting them together for improved bug localization[C]//Proceedings of the 22nd International Conference on Program Comprehension. ACM, 2014: 53-63.
8. Ye X, Bunescu R, Liu C. Learning to rank relevant files for bug reports using domain knowledge[C]//Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. ACM, 2014: 689-699.
9. Rao S, Medeiros H, Kak A. An incremental update framework for efficient retrieval from software libraries for bug localization[C]//Reverse Engineering (WCRE), 2013 20th Working Conference on. IEEE, 2013: 62-71.
10. Sisman B, Kak A C. Incorporating version histories in information retrieval based bug localization[C]//Proceedings of the 9th IEEE Working Conference on Mining Software Repositories. IEEE Press, 2012: 50-59.
11. Almhana R, Mkaouer W, Kessentini M, et al. Recommending relevant classes for bug reports using multi-objective search[C]//Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering. ACM, 2016: 286-295.

12. Rao S, Kak A. Retrieval from software libraries for bug localization: a comparative study of generic and composite text models[C]//Proceedings of the 8th Working Conference on Mining Software Repositories. ACM, 2011: 43-52.
13. Nguyen A T, Nguyen T T, Al-Kofahi J, et al. A topic-based approach for narrowing the search space of buggy files from a bug report[C]//Automated Software Engineering (ASE), 2011 26th IEEE/ACM International Conference on. IEEE, 2011: 263-272.
14. Wang S, Lo D, Lawall J. Compositional vector space models for improved bug localization[C]//Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on. IEEE, 2014: 171-180.
15. Saha R K, Lawall J, Khurshid S, et al. On the effectiveness of information retrieval based bug localization for c programs[C]//Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on. IEEE, 2014: 161-170.
16. Zhou J, Zhang H, Lo D. Where should the bugs be fixed?-more accurate information retrieval-based bug localization based on bug reports[C]//Proceedings of the 34th International Conference on Software Engineering. IEEE Press, 2012: 14-24.
17. Thomas S W, Nagappan M, Blostein D, et al. The impact of classifier configuration and classifier combination on bug localization[J]. IEEE Transactions on Software Engineering, 2013, 39(10): 1427-1443.
18. Wong C P, Xiong Y, Zhang H, et al. Boosting bug-report-oriented fault localization with segmentation and stack-trace analysis[C]//Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on. IEEE, 2014: 181-190.
19. Zhang Y, Lo D, Xia X, et al. Inferring links between concerns and methods with multi-abstraction vector space model[C]//Software Maintenance and Evolution (ICSME), 2016 IEEE International Conference on. IEEE, 2016: 110-121.
20. Lukins SK, Kraft NA, Etzkorn LH. Source code retrieval for bug localization using latent dirichlet allocation. In Reverse Engineering, 2008. WCRE'08. 15th Working Conference on 2008 Oct 15 (pp. 155-164). IEEE.
21. Lam AN, Nguyen AT, Nguyen HA, Nguyen TN. Combining deep learning with information retrieval to localize buggy files for bug reports (n). In Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on 2015 Nov 9 (pp. 476-481). IEEE.
22. Chaparro O, Florez JM, Marcus A. Using Observed Behavior to Reformulate Queries during Text Retrieval-based Bug Localization. InSoftware Maintenance and Evolution (ICSME), 2017 IEEE International Conference on 2017 Sep 17 (pp. 376-387). IEEE.

2) Bug Triaging

(These techniques process a bug report, and recommend the most appropriate developers to fix the bug.)

1. Bortis G, van der Hoek A. Porchlight: A tag-based approach to bug triaging[C]//Software Engineering (ICSE), 2013 35th International Conference on. IEEE, 2013: 342-351.
2. Xia X, Lo D, Wang X, et al. Accurate developer recommendation for bug resolution[C]//Reverse engineering (WCRE), 2013 20th working conference on. IEEE, 2013: 72-81.
3. Anvik J, Murphy G C. Reducing the effort of bug report triage: Recommenders for development-oriented decisions[J]. ACM Transactions on Software Engineering and Methodology (TOSEM), 2011, 20(3): 10.
4. Wang S, Zhang W, Wang Q. FixerCache: Unsupervised caching active developers for diverse bug triage[C]//Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. ACM, 2014: 25.
5. Linares-Vásquez M, Hossen K, Dang H, et al. Triaging incoming change requests: Bug or commit history, or code authorship?[C]//Software Maintenance (ICSM), 2012 28th IEEE International Conference on. IEEE, 2012: 451-460.

6. Shokripour R, Anvik J, Kasirun Z M, et al. Why so complicated? simple term filtering and weighting for location-based bug report assignment recommendation[C]//Proceedings of the 10th Working Conference on Mining Software Repositories. IEEE Press, 2013: 2-11.
7. Naguib H, Narayan N, Brügge B, et al. Bug report assignee recommendation using activity profiles[C]//Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on. IEEE, 2013: 22-30.
8. Tian Y, Wijedasa D, Lo D, et al. Learning to rank for bug report assignee recommendation[C]//Program Comprehension (ICPC), 2016 IEEE 24th International Conference on. IEEE, 2016: 1-10.
9. Tamrawi A, Nguyen T T, Al-Kofahi J M, et al. Fuzzy set and cache-based approach for bug triaging[C]//Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering. ACM, 2011: 365-375.
10. Bhattacharya P, Neamtiu I. Fine-grained incremental learning and multi-feature tossing graphs to improve bug triaging[C]//Software Maintenance (ICSM), 2010 IEEE International Conference on. IEEE, 2010: 1-10.
11. Jonsson L, Borg M, Broman D, Sandahl K, Eldh S, Runeson P. Automated bug assignment: Ensemble-based machine learning in large scale industrial contexts. Empirical Software Engineering. 2016 Aug 1;21(4):1533-78.
12. Bhattacharya P, Neamtiu I, Shelton CR. Automated, highly-accurate, bug assignment using machine learning and tossing graphs. Journal of Systems and Software. 2012 Oct 1;85(10):2275-92.
13. Shokripour R, Anvik J, Kasirun ZM, Zamani S. A time-based approach to automatic bug report assignment. Journal of Systems and Software. 2015 Apr 1;102:109-22.
14. Cavalcanti YC, do Carmo Machado I, Neto PA, de Almeida ES. Towards semi-automated assignment of software change requests. Journal of Systems and Software. 2016 May 1;115:82-101.
15. Zhang T, Chen J, Yang G, Lee B, Luo X. Towards more accurate severity prediction and fixer recommendation of software bugs. Journal of Systems and Software. 2016 Jul 1;117:166-84.
16. Sun X, Yang H, Xia X, Li B. Enhancing developer recommendation with supplementary information via mining historical commits. Journal of Systems and Software. 2017 Dec 1;134:355-68.
17. Matter D, Kuhn A, Nierstrasz O. Assigning bug reports using a vocabulary-based expertise model of developers. In Mining Software Repositories, 2009. MSR'09. 6th IEEE International Working Conference on 2009 May 16 (pp. 131-140). IEEE.
18. Jeong G, Kim S, Zimmermann T. Improving bug triage with bug tossing graphs. InProceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering 2009 Aug 24 (pp. 111-120). ACM.
19. Anvik J, Hiew L, Murphy GC. Who should fix this bug?. InProceedings of the 28th international conference on Software engineering 2006 May 28 (pp. 361-370). ACM.
20. Xia X, Lo D, Ding Y, Al-Kofahi JM, Nguyen TN, Wang X. Improving automated bug triaging with specialized topic model. IEEE Transactions on Software Engineering. 2017 Mar 1;43(3):272-97.
21. Xuan J, Jiang H, Ren Z, Zou W. Developer prioritization in bug repositories. In Software Engineering (ICSE), 2012 34th International Conference on 2012 Jun 2 (pp. 25-35). IEEE.

3) Duplicate/Similar bug detection

(These techniques detect duplicate/similar bug reports in bug repositories.)

1. Liu K, Tan H B K, Zhang H. Has this bug been reported?[C]//Reverse Engineering (WCRE), 2013 20th Working Conference on. IEEE, 2013: 82-91.

2. Amoui M, Kaushik N, Al-Dabbagh A, et al. Search-based duplicate defect detection: an industrial experience[C]//Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on. IEEE, 2013: 173-182.
3. Alipour A, Hindle A, Stroulia E. A contextual approach towards more accurate duplicate bug report detection[C]//Proceedings of the 10th Working Conference on Mining Software Repositories. IEEE Press, 2013: 183-192.
4. Rocha H, Valente M T, Marques-Neto H, et al. An empirical study on recommendations of similar bugs[C]//Software Analysis, Evolution, and Reengineering (SANER), 2016 IEEE 23rd International Conference on. IEEE, 2016, 1: 46-56.
5. Nguyen A T, Nguyen T T, Nguyen T N, et al. Duplicate bug report detection with a combination of information retrieval and topic modeling[C]//Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering. ACM, 2012: 70-79.
6. Sun C, Lo D, Khoo S C, et al. Towards more accurate retrieval of duplicate bug reports[C]//Automated Software Engineering (ASE), 2011 26th IEEE/ACM International Conference on. IEEE, 2011: 253-262.
7. Sun C, Lo D, Wang X, et al. A discriminative model approach for accurate duplicate bug report retrieval[C]//Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1. ACM, 2010: 45-54.
8. Hindle A, Alipour A, Stroulia E. A contextual approach towards more accurate duplicate bug report detection and ranking. Empirical Software Engineering. 2016 Apr 1;21(2):368-410.
9. Wang X, Zhang L, Xie T, Anvik J, Sun J. An approach to detecting duplicate bug reports using natural language and execution information. In Software Engineering, 2008. ICSE'08. ACM/IEEE 30th International Conference on 2008 May 10 (pp. 461-470). IEEE.
10. Runeson P, Alexandersson M, Nyholm O. Detection of duplicate defect reports using natural language processing. In Proceedings of the 29th international conference on Software Engineering 2007 May 24 (pp. 499-510). IEEE Computer Society.
11. Aggarwal K, Rutgers T, Timbers F, Hindle A, Greiner R, Stroulia E. Detecting duplicate bug reports with software engineering domain knowledge. In2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER).
12. Lin MJ, Yang CZ, Lee CY, Chen CC. Enhancements for duplication detection in bug reports with manifold correlation features. Journal of Systems and Software. 2016 Nov 1;121:223-33.
13. Deshmukh J, Podder S, Sengupta S, Dubash N. Towards Accurate Duplicate Bug Retrieval Using Deep Learning Techniques. InSoftware Maintenance and Evolution (ICSME), 2017 IEEE International Conference on 2017 Sep 17 (pp. 115-124). IEEE.

4) Bug categorization

(These techniques process a bug report, and classify it into different categories (e.g., repro- ducible bug report or not, invalid bug report or not, bug fixing request or feature request, security bug report or not, etc.))

1. Thung F, Lo D, Jiang L. Automatic defect categorization[C]//Reverse Engineering (WCRE), 2012 19th Working Conference on. IEEE, 2012: 205-214.
2. Zanetti M S, Scholtes I, Tessone C J, et al. Categorizing bugs with social networks: a case study on four open source software communities[C]//Software Engineering (ICSE), 2013 35th international conference on. IEEE, 2013: 1032-1041.
3. Thung F, Le X B D, Lo D. Active semi-supervised defect categorization[C]//Proceedings of the 2015 IEEE 23rd International Conference on Program Comprehension. IEEE Press, 2015: 60-70.
4. Valdivia Garcia H, Shihab E. Characterizing and predicting blocking bugs in open source projects[C]//Proceedings of the 11th working conference on mining software repositories. ACM, 2014: 72-81.
5. Gegick M, Rotella P, Xie T. Identifying security bug reports via text mining: An industrial case study[C]//Mining software repositories (MSR), 2010 7th IEEE working conference on. IEEE, 2010: 11-20.

6. Choetkiertikul M, Dam H K, Tran T, et al. Characterization and prediction of issue-related risks in software projects[C]//Proceedings of the 12th Working Conference on Mining Software Repositories. IEEE Press, 2015: 280-291.
7. Tan L, Liu C, Li Z, Wang X, Zhou Y, Zhai C. Bug characteristics in open source software. Empirical Software Engineering. 2014 Dec 1;19(6):1665-705.
8. Wang J, Wang S, Cui Q, Wang Q. Local-based active classification of test report to assist crowdsourced testing. In Automated Software Engineering (ASE), 2016 31st IEEE/ACM International Conference on 2016 Sep 3 (pp. 190-201). IEEE.
9. Hooimeijer P, Weimer W. Modeling bug report quality. In Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering 2007 Nov 5 (pp. 34-43). ACM.
10. Fan Q, Yu Y, Yin G, Wang T, Wang H. Where Is the Road for Issue Reports Classification Based on Text Mining?. In Empirical Software Engineering and Measurement (ESEM), 2017 ACM/IEEE International Symposium on 2017 Nov 9 (pp. 121-130). IEEE.

5) Bug fixing time prediction

(These techniques process a bug report, and predict how long it will take to fix the bug.)

1. Zhang H, Gong L, Versteeg S. Predicting bug-fixing time: an empirical study of commercial software projects[C]//Proceedings of the 2013 international conference on software engineering. IEEE Press, 2013: 1042-1051.
2. Kikas R, Dumas M, Pfahl D. Using dynamic and contextual features to predict issue lifetime in GitHub projects[C]//Proceedings of the 13th International Conference on Mining Software Repositories. ACM, 2016: 291-302.
3. Guo P J, Zimmermann T, Nagappan N, et al. Characterizing and predicting which bugs get fixed: an empirical study of Microsoft Windows[C]//Software Engineering, 2010 ACM/IEEE 32nd International Conference on. IEEE, 2010, 1: 495-504.
4. Raja U. All complaints are not created equal: text analysis of open source software defect reports. Empirical Software Engineering. 2013 Feb 1;18(1):117-38.
5. Assar S, Borg M, Pfahl D. Using text clustering to predict defect resolution time: a conceptual replication and an evaluation of prediction accuracy. Empirical Software Engineering. 2016 Aug 1;21(4):1437-75.
6. Choetkiertikul M, Dam HK, Tran T, Ghose A. Predicting the delay of issues with due dates in software projects. Empirical Software Engineering. 2017 Jan 19:1-41.
7. Weiss C, Premraj R, Zimmermann T, Zeller A. How long will it take to fix this bug?. In Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on 2007 May 20. IEEE. 8 pages.
8. Hewett R, Kijsanayothin P. On modeling software defect repair time. Empirical Software Engineering. 2009 Apr 1;14(2):165.

6) Bug severity/priority prediction

(These techniques process a bug report, and predict its severity/priority.)

1. Lamkanfi A, Demeyer S, Giger E, et al. Predicting the severity of a reported bug[C]//Mining Software Repositories (MSR), 2010 7th IEEE Working Conference on. IEEE, 2010: 1-10.
2. Tian Y, Lo D, Sun C. Information retrieval based nearest neighbor classification for fine-grained bug severity prediction[C]//Reverse Engineering (WCRE), 2012 19th Working Conference on. IEEE, 2012: 215-224.
3. Tian Y, Lo D, Sun C. DRONE: Predicting Priority of Reported Bugs by Multi-factor Analysis[C]//ICSM. 2013: 200-209.
4. Tian Y, Lo D, Xia X, Sun C. Automated prediction of bug report priority using multi-factor analysis. Empirical Software Engineering. 2015 Oct 1;20(5):1354-83.
5. Zhang T, Chen J, Yang G, Lee B, Luo X. Towards more accurate severity prediction and fixer recommendation of software bugs. Journal of Systems and Software. 2016 Jul 1;117:166-84.

6. Menzies T, Marcus A. Automated severity assessment of software defect reports. In Software Maintenance, 2008. ICSM 2008. IEEE International Conference on 2008 Sep 28 (pp. 346-355). IEEE.
7. Feng Y, Chen Z, Jones JA, Fang C, Xu B. Test report prioritization to assist crowdsourced testing. InESEC/SIGSOFT FSE 2015 Aug 30 (pp. 225-236).
8. Feng Y, Jones JA, Chen Z, Fang C. Multi-objective test report prioritization using image understanding. In Automated Software Engineering (ASE), 2016 31st IEEE/ACM International Conference on 2016 Sep 3 (pp. 202-213). IEEE.

7) Bug-commit Linking

(These techniques aim to link bug reports with bug fixing commits or bug inducing commits. With these techniques, developers can better understand which commits fix the bug and why/how/when the bug is introduced.)

1. Wu R, Zhang H, Kim S, et al. Relink: recovering links between bugs and changes[C]//Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering. ACM, 2011: 15-25.
2. Schermann G, Brandtner M, Panichella S, et al. Discovering loners and phantoms in commit and issue data[C]//Program Comprehension (ICPC), 2015 IEEE 23rd International Conference on. IEEE, 2015: 4-14.
3. Le T D B, Linares-Vásquez M, Lo D, et al. Rclinker: Automated linking of issue reports and commits leveraging rich contextual information[C]//Program Comprehension (ICPC), 2015 IEEE 23rd International Conference on. IEEE, 2015: 36-47.
4. Nguyen A T, Nguyen T T, Nguyen H A, et al. Multi-layered approach for recovering links between bug reports and fixes[C]//Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering. ACM, 2012: 63.
5. Davies S, Roper M, Wood M. A preliminary evaluation of text-based and dependency-based techniques for determining the origins of bugs[C]//Reverse Engineering (WCRE), 2011 18th Working Conference on. IEEE, 2011: 201-210.
6. Bachmann A, Bird C, Rahman F, et al. The missing links: bugs and bug-fix commits[C]//Proceedings of the eighteenth ACM SIGSOFT international symposium on Foundations of software engineering. ACM, 2010: 97-106.
7. Kim S, Zimmermann T, Pan K, James Jr E. Automatic identification of bug-introducing changes. In Automated Software Engineering, 2006. ASE'06. 21st IEEE/ACM International Conference on 2006 Sep 18 (pp. 81-90). IEEE.

8) Bug Report Completion/Refinement

(These techniques aim to generate a high-quality bug report. Some of these techniques auto-matically generate a bug report when software crashes. Some others help to make a better- quality bug report by enriching/modifying an existing one.)

1. White M, Linares-Vásquez M, Johnson P, et al. Generating reproducible and replayable bug reports from android application crashes[C]//Program Comprehension (ICPC), 2015 IEEE 23rd International Conference on. IEEE, 2015: 48-59.
2. Moran K, Linares-Vásquez M, Bernal-Cárdenas C, et al. Automatically discovering, reporting and reproducing android application crashes[C]//Software Testing, Verification and Validation (ICST), 2016 IEEE International Conference on. IEEE, 2016: 33-44.
3. Moran K, Linares-Vásquez M, Bernal-Cárdenas C, et al. Auto-completing bug reports for android applications[C]//Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. ACM, 2015: 673-686.
4. Tao Zhang, Jiachi Chen, He Jiang, Xiapu Luo and Xin Xia. *Bug Report Enrichment with Application of Automated Fixer Recommendation*. in ICPC 2017.
5. Thomas Zimmermann, Rahul Premraj, Nicolas Bettenburg, Sascha Just, Adrian Schröter, Cathrin Weiss: What Makes a Good Bug Report? IEEE Trans. Software Eng. 36(5): 618-643 (2010).

6. Chaparro O, Lu J, Zampetti F, Moreno L, Di Penta M, Marcus A, Bavota G, Ng V. Detecting missing information in bug descriptions. In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering 2017 Aug 21 (pp. 396-407). ACM.

9) Bug report summarization/visualization

(These techniques process a bug report, and summarize it into a much shorter form. Some of these techniques also help developers better navigate/understand bug reports through visualization.)

1. Rastkar S, Murphy G C, Murray G. Automatic summarization of bug reports[J]. IEEE Transactions on Software Engineering, 2014, 40(4): 366-380.
2. Rastkar S, Murphy G C, Murray G. Summarizing software artifacts: a case study of bug reports[C]//Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering. ACM, 2010: 505-514.
3. Mani, Senthil, et al. Ausum: approach for unsupervised bug report summarization. Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering. ACM, 2012.
4. Lotufo R, Malik Z, Czarnecki K. Modelling the 'hurried'bug report reading process to summarize bug reports. Empirical Software Engineering. 2015 Apr 1;20(2):516-48.
5. Baysal O, Holmes R, Godfrey M W. No issue left behind: Reducing information overload in issue tracking[C]//Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering. ACM, 2014: 666-677.

10) Re-opend bug prediction

(These techniques process a closed bug report, and predict whether it is likely to be re-opened.)

1. Zimmermann T, Nagappan N, Guo P J, et al. Characterizing and predicting which bugs get reopened[C]//Proceedings of the 34th International Conference on Software Engineering. IEEE Press, 2012: 1074-1083.
2. Xia X, Lo D, Wang X, et al. A comparative study of supervised learning algorithms for re-opened bug prediction[C]//Software Maintenance and Reengineering (CSMR), 2013 17th European Conference on. IEEE, 2013: 331-334.
3. Shihab E, Ihara A, Kamei Y, et al. Predicting re-opened bugs: A case study on the eclipse project[C]//Reverse Engineering (WCRE), 2010 17th Working Conference on. IEEE, 2010: 249-258.
4. Shihab E, Ihara A, Kamei Y, Ibrahim WM, Ohira M, Adams B, Hassan AE, Matsumoto KI. Studying re-opened bugs in open source software. Empirical Software Engineering. 2013 Oct 1;18(5):1005-42.

11) Bug number prediction

(These techniques predict the number of bug reports in different state (i.e., NEW, ASSIGNED, etc.))

1. Wang J, Zhang H. Predicting defect numbers based on defect state transition models[C]//Empirical Software Engineering and Measurement (ESEM), 2012 ACM-IEEE International Symposium on. IEEE, 2012: 191-200.

12) Bug reporting latency prediction

(These techniques predict whether a bug report can be reported earlier.)

1. Thung F, Lo D, Jiang L, et al. When would this bug get reported?[C]//Software Maintenance (ICSM), 2012 28th IEEE International Conference on. IEEE, 2012: 420-429.

13) Bug report topic extraction

(These techniques using specific methods such as LDA to extract topics from bug reports.)

1. Hindle A, Bird C, Zimmermann T, Nagappan N. Do topics make sense to managers and developers?. Empirical Software Engineering. 2015 Apr 1;20(2):479-515.

14) Bug repair

(These techniques recommend candidate repairs from bug reports.)

1. Liu C, Yang J, Tan L, et al. R2Fix: Automatically generating bug fixes from bug reports[C]//Software Testing, Verification and Validation (ICST), 2013 IEEE Sixth International Conference on. IEEE, 2013: 282-291.

15) Determining implementation expertise from bug reports

(these techniques determine developers' implementation expertise from bug reports.)

1. Anvik J, Murphy GC. Determining implementation expertise from bug reports. InProceedings of the Fourth International Workshop on Mining Software Repositories 2007 May 20 (p. 2). IEEE Computer Society.