# On the Use of Hidden Markov Model to Predict the Time to Fix Bugs

Mayy Habayeb*, Syed Shariyar Murtaza†, Andriy Miranskyy‡, Ayse Basar Bener§
*†§ Department of Mechanical and Industrial Engineering, Data Science Lab, Ryerson University, Toronto, Canada
‡ Department of Computer Science, Data Science Lab, Ryerson University, Toronto, Canada
mayy.habayeb@gmail.com, syed.shariyar@gmail.com, avm@ryerson.ca, ayse.bener@ryerson.ca

## KEYWORDS

Bug repositories, Temporal activities, Time to fix a bug, Hidden Markov model.

A significant amount of time is spent by software developers in investigating bug reports. It is useful to indicate when a bug report will be closed, since it would help software teams to prioritise their work. Several studies have been conducted to address this problem in the past decade. Most of these studies have used the frequency of occurrence of certain developer activities as input attributes in building their prediction models. However, these approaches tend to ignore the temporal nature of the occurrence of these activities. In this paper, a novel approach using Hidden Markov models (HMMs) and temporal sequences of developer activities is proposed. The approach is empirically demonstrated in a case study using eight years of bug reports collected from the Firefox project. We provide additional details below. In a software bug repository, recorded developer activities occur sequentially. For example, activity $C$ (a certain person has been copied on the bug report) is followed by activity $A$ (bug confirmed and assigned to a named developer), which in turn is followed by activity $Z$ (bug reached status resolved). Additional piece of information is developers' level of expertise, such as novice ($N$), intermediate ($M$), or experienced ($E$), at the time of report creation. We combine these data together to produce a sequence of temporal activities associated with bug reports in the Firefox bug repository.

Sequences can also be lengthy; the length of the sequence and variability of activities may indicate active efforts and progress toward resolving the bug report.

We analysed the temporal sequences generated for the eight-year period and obtained two major findings: 1) most of the bug reports had similar temporal sequences of activities; and 2) many bugs start as reported (in an unconfirmed state) and move immediately into the resolved state without passing through additional stages (e.g., assignment, development, and code review). This implies that not all the activities of developers and testers are tracked by the bug repository. These activities are hidden from us. These findings inspired us to use the HMMs. In our approach, we used a heuristic based on the assumption that similar sequences of developer activities will take similar periods of time.

Our objective was to predict whether the bug report will take longer time (slow) to fix beyond a certain threshold or a shorter time (fast) to fix. To achieve this, we split the set of sequences (extracted from the bug reports) into two subsets: those that require a total number of days to fix the bug below a certain threshold (fast) and the remaining ones (slow). A threshold of two months (approximately the median duration of the time-to-fix in the dataset) was chosen. We train individual HMMs on each of the two subsets of sequences.

We performed several experiments to demonstrate the usability of our approach. In our first experiment, we focused on demonstrating the ability of classification on retrieving various fixed lengths of temporal sequences of developer activities for bugs in the test set. The results showed that when our approach is applied to the first three to four initial activities happening on the bug report, it is able to correctly classify bug reports as fast and slow with a precision of 77.12% and recall of 71.58%.

In the second experiment, we focused on classification of temporal sequences of bugs based on the developer activities recorded up to a fixed number of days; e.g., first week after initial bug report. Our results show 58.44% recall and 80.80% precision for classification of bug reports in this experiment.

In the third experiment, we trained HMMs on bugs of previous years and tested on bugs of future years. For example, we trained our model using the bug report sequences reported in year 2013 and then tested our model on bug report sequences of year 2014. Our approach achieved a precision of 82.98% with a recall rate of 66.65%. We also compared our proposed approach with the state of the art technique in the context of our case study. Our approach results in approximately 33% higher F-measure than that of the contemporary techniques.

In summary, our three contributions are as follows. 1) A technique to create a temporal dataset of activities that occurred during the life cycle of bugs in a bug repository. 2) A Firefox dataset of temporal activities, available on-line at https://drive.google.com/drive/u/1/folders/0B5TJwohpS9LzcHc3NldtdjZQRFk. 3) An approach to predict the time to fix bugs modelling temporal activities using the HMMs. For details of our work please see [1].


*ICSE '18, May 27-June 3, 2018, Gothenburg, Sweden*

ACM ISBN 978-1-4503-5638-1/18/05.
https://doi.org/10.1145/3180155.3182522

## REFERENCES

[1] M. Habayeb, S. S. Murtaza, A. Miranskyy, and A. B. Bener. 2017. On the Use of Hidden Markov Model to Predict the Time to Fix Bugs. *IEEE Transactions on Software Engineering* (2017). https://doi.org/10.1109/TSE.2017.2757480