ENHANCED BUG TRAIGING WITH DATA TRUNCATION

KRISHNAVENI V.L,KARTHIK.M

Abstract- Software bugs being an inevitable substance, which is growing up daily as a distractor for the software field. Assigning an efficient and well experienced developer for the bugs are a hazardous task. Bug traiging is a processing that is defined in a manner that to properly select a developer who can handle and fix the bug in an effective manner. Historical dataset plays a major role as a supporter in tariging process, that is we could use it as a reference when a new bug arrives. As per the traditional mannerism, the human traiger does the work of assigning the developer, but it overpowered with automatic startegy. Automatic bug traiging with truncated data source could pave the way much more easier and thus could select the accurate developer for the processing. The Instance and Feature Selection combination help in the case of reduction data. Large amount of data can be truncated throughout. Finally the classifier is demanded for the classification. The most efficient binary classifier (Naive Baye's), moved on to Multi Classifier (Random Forest) is used for classification. Finally it will lead to the assignment of most accurate developer for the present software bug.

Keywords:- Data, Features, Bug, Instance, Traiging.

1. INTRODUCTION

Generally software companies spend a good sum of amount for the repairment of bug or to be fixed it with. Multi or vast software projects have bug repository that keeps all the information related to the bugs and is very well maintained for further processing. In the case of bug repository, each software bug has its own report and is also known by the name bug data. It keeps track of textual information of the bug and the updates on the basis of the status of bug fixing, which is available in historical bug data set accordingly.

Researchers proposed various techniques to have a look into the bug-report analysis. Certain open source software projects keep on updating about the defects and issues facing in the field as a record, also keep a suggestion about possible modification with enhancement. Due to the arising of the everyday bug, the maintenance is getting much tougher.

Two challenges that comes face to face is that namely the huge amount of data and less quality. Manual bug triaging by a human bug triager is a vast process and the same time it may find difficulty due to the errors which occurs by the arrival of huge amount of bug data and less availability of developers who has an accurate knowledge of the bug to be fixed with. In some former methods, the manual assigning process is does, whenever the bug is found as such. Those developers status is stored in historical recored as item assigned.

The data truncation holds for the betterment of traiging. The mingled techniques such as instance selection along with that of feature selection is used to truncate the dimensions of bug and the data. The new version consist with similar meaning of old one with less data amount.

The application of the processing is done in its evident way is that for the truncation purpose and the quality maintenance afford the feature and instance process of selecting, prediction is done further and for that classifier is used. Here both Naïve and Random techniques are used. Random forest is also used for prediction, because RF itself is a multiclassifier and provide more accurate and better result, when it comes to the case of assigning developer. Comparative result is also provided.

2. LITERATURE SURVEY

In [1] they had a deeper look on to the malfunctioning of the web scripts as well as the crashes that occurs along with the functioning. A derived solution for that serious approach was Test generation in dynamic for Automatic test is done, then captures logical on inputs, reduction in test failure. At the same time it is simple and safe too. An Apollo tool with PHP Programming is specified.

Krishnaveni V.I is currently pursuing masters degree program in computer science and engineering in Mohandas College of Engineering, Kerala University, Kerala, India, PH-+91928190776. E-mail:anjali.ditesai@gmail.com

Karthik.M is working as Assistant Professor of computer science and engineering in Mohandas College of Engineering, Kerala University, Kerala, India, PH-+919496327831. E-mail: karthikcpta@gmail.com
In [2] they mentioned bug triaging and also gave a deep focusing look into the matter as a whole, which is of “Bug Triage and Resolution Practices”, mostly the analysis is done on the basis of the root cause analysis of reassignment and among those bugs which were re-opened. Length variation occurs in data, it is printed and splitted. They also planned to perform a sorting with card, keeping in touch with response of interview as a medium to lessen those level and also keep focusing to identify the common themes in the participant’s feedback methodology.

In [3], Bug tossing history approach is mostly concentrated over here by markov chain methodization. Primarily it gives clear idea about networking structure. It essentializes the probability of tossing between developers from the history which is available in bug tracking systems.

In [4] study of optimization and the accuracy is done and it leads to a solution which leads to the recommendation. First, reformulate the problem as an optimization problem of both accuracy and cost then Second, CBF is coping up with the currently using one with a (CF) was the best-chosed option, and modify the recommendation quality of either approach alone.

In [5], the approach does is semi-supervised which is far better when compared to the previous methods, The mixed usage of naive classifier and the expectation in the maximization has been considered, both labeled and unlabeled bug reports. The techniques offer some contributions such as bug triaging by semi-supervised approach and weight recommendation list is maintained.

In [6] they addressed the data truncation and quality. The triaging of the bug lights on to the matter of assignment of developer who is well capable of handling it so. For e.g., the new bug reports or will-not-fix bug reports.

The mingled techniques of instance election with feature selection for the job of reduction in data scaling on the behalf dimension of bug and data is given for reference. To getting the order of applying selections, the accurate extracted information of datas from the historically maintaining data is considered and create a model for prediction.

**3. PROPOSED SYSTEM / MODULE DESCRIPTION**

Manual Bug fixing is not worth of time usage and never provides accurate result. To come up with the difficulties this proposed system is been introduced. There is problem of getting an exact bug solution according to the domain as per. According to the previous approach, the automatic triaging is done with reduced set but the naive bayes classifier couldn’t fulfill the multi classification and accuracy in developer assignment. To pay attention towards a precise prediction, existing system instance selection and feature selection for reducing bug dataset is done. Furtherly does clustering in bug data inorder of reporter, developer etc. Finally classifies with Random forest algorithm in prediction, for the improvement of data reduction quality by the means of result when compares to that of existing system and get domain wise bug solution in automatic bug triaging.
**A. Architecture**

The architecture is divided into 3 parts as Historical data repository as primary stage, next moves on with the selection processing and prediction orders. Ends with the classifier application for developer assignment. Both data truncation as well as the redundancy removal is paid with more concentration too.

a) Assignment of a apt developer for the newly reported bug in maintenance is generally mentioned as traiging of bug. The Repository will provide the status whether it could be rectified or not and also in bug repository previous bug data details are also present for the reference purpose [5].

b) Quality improvement of data is does by applying instance as well as the feature selectioning. Finally we will be awarded with atmost quality data that too a truncated one.

c) Original data set is classified with the upcoming new bug report and this job is carried out by none other than the classifier. According to the performance in solving the bugs, the priority of the developer status is also maintained. Classifier is applied to the data which is available with us that was truncated before itself through the Is and Fs techniques [9]. If the expert solved only very few bugs and remains idle then the expert is removed from that priority list too.

**B. Algorithm**

**Algorithm 1. Data reduction based on FS → IS**

| Input: | training set $T$ with $n$ words and $m$ bug reports, $\text{reduction order } FS \rightarrow IS$, final number $n_F$ of words, final number $m_j$ of bug reports, |
| Output: | reduced data set $T_{f1}$ for bug triage |

1. apply $FS$ to $n$ words of $T$ and calculate objective values for all the words;
2. select the top $n_F$ words of $T$ and generate a training set $T_{f1}$;
3. apply $IS$ to $m_j$ bug reports of $T_{f1}$;
4. terminate $IS$ when the number of bug reports is equal to or less than $m_j$ and generate the final training set $T_{f1}$.

- **IS(Instance selection)** – process of developing much needed subset of instances removing stem words, bag of words etc (i.e., bug reports filtration as well bug data).
- **FS(feature selection)** which focus towards the miniature form of features (i.e., filtering by words).

IS and FS is mingled so that we could achieve both quality data in a miniature form. Before application was done separately but it never fulfilled neither accuracy nor truncation. Thus here it is used together to achieve best. Reduction order is an important criteria for prediction.
4. SYSTEM ANALYSIS/ EXPERIMENTAL RESULTS

The processing is divided into 4 sections:-

1. Once a new bugset is provided, we can have a look on to the developer status from bug repository. If that kind of bug has never been handles with any of the developer then we shall move further.

2. In the 2nd section, we can apply Is and Fs for redundancy check and as well as the dataset truncation. At the end of this processing we will be awarded with truncated data with a good quality.

3. Next we can identify through clustering, how many bugs are rectified by the developer’s. It will give complete status about the bugs to the admin or the reporter and there by we will get knowledge on the bug status. Move on to prediction orders too.

4. Finally we shall apply classifier for prediction orders with the help of historical data set and the data which is available with us after those earlier processing. Both Naive and Random forest algorithm is used and does the comparison too. random forest is applied to get more precision while assigning the developer.

A. NAIVE BAYE’S

A well familiar one used for features/predictors in a learning problem. Maximum processing or training is does expression of closureness while evaluating. It probably leap, instead of approximating with high costage like other types of classifiers as such.

1. Learning Phase: Given a training set of F features and L classes.

2. Testing Phase: Look up tables to assign the label

B. RANDOM FOREST

Random forest technique pves the individual tree class output mode. Mainly a approach of tree with decision processing is does.

1. Create training set:-

Random n sample is selected with replacement of N available sample tainee.

2. Create decision rule at node of each tree:-
   - Randomly process and select m features for the decision at that node.
   - Select feature (from m available) that maximizes information gain in the set of training samples.

3. Tree are grown completely and not at all pruned.

Fig 4. Naïve Baye’s Output

Fig 5. Random Forest Output
5. CONCLUSION

Software world faces much difficulty in the case of bug maintenance, thereby it led to drastic loss in time and money. Muddled usage of feature and instance selection pave the benefit in providing quality in data and the truncation of it. Innovative usage of Classifier for prediction order award with more accuracy and precision in assigning the developer for a newly reported bug.

References


