Paper Review Presentation
Paper Title: Who Should Fix This Bug?

Marzia Zaman (101018327)

September 22, 2016

Problem Statement

- For a large open source software project triage process may take a long time

  "Consider the case of the Eclipse open source project over a four month period (January 1, 2005 to April 30, 2005) when 3426 reports were filed, averaging 29 reports per day."

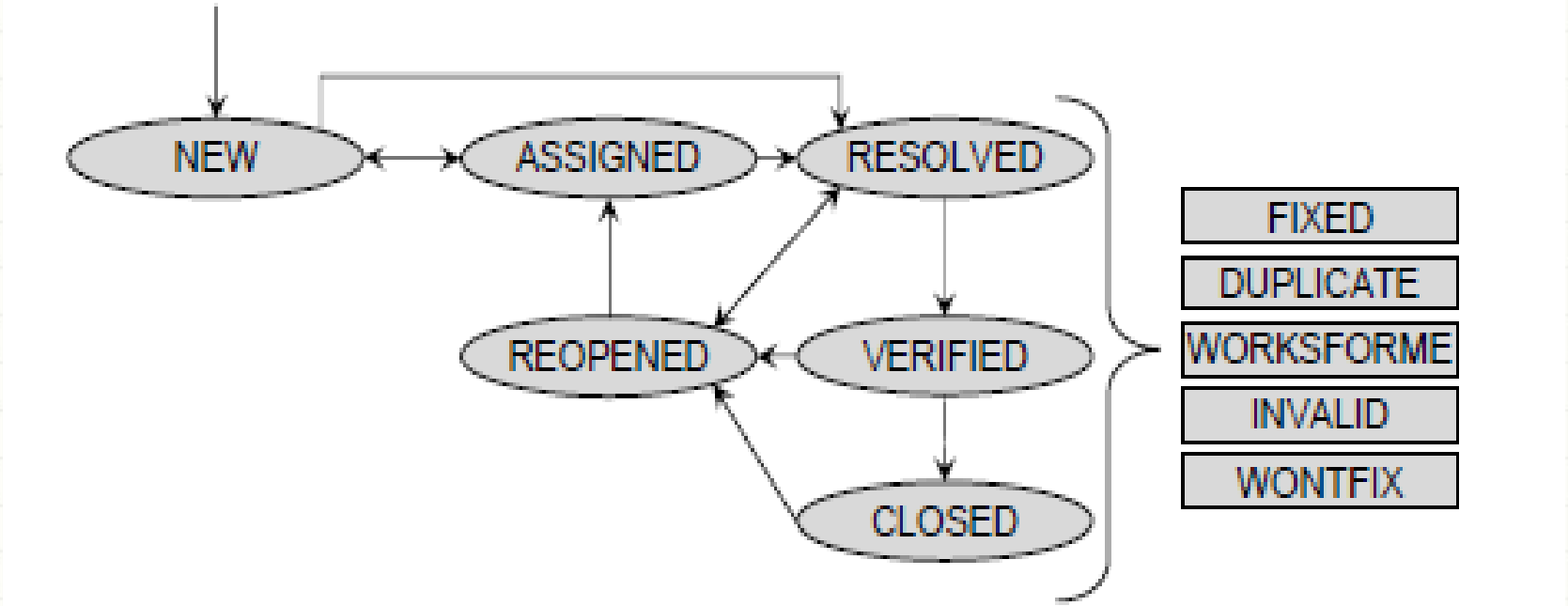
  "Everyday, almost 300 bugs appear that need triaging. This is far too much for only the Mozilla programmers to handle"

- The bug assignment may not be as accurate as one wants it to be – the triager may not have adequate knowledge about the software and its developers to decide the appropriate person

Objective

- To reduce the triage time by semi-automating the assignment process
- Make the assignment process more accurate

Bug Report Life Cycle



Projects under Study

- Three Open Source Software Projects:
  - Eclipse
  - Mozilla Firefox
  - Gcc
- Number of bugs (daily)

	Around Release			After Release		
	Mean	Min	Max	Mean	Min	Max
Eclipse	48	1	192	13	1	124
Firefox	8	1	37	5	1	37

- Source of Data : Bugzilla

Data Set

- The training sets included 8655 reports for Eclipse and 9752 for Firefox between September 1, 2004 and May 31, 2005.
- Test sets consisted of 170 reports for Eclipse and 22 reports for Firefox from May 2005

Proposed Methodology

- Recommender System based on classification
- Feature Vectors are created by mining text from Bugzilla report
  - Summary
  - Description

Validation Method

$$Precision = \frac{\#\ of\ appropriate\ recommendations}{\#\ of\ recommendations\ made}$$

$$Recall = \frac{\#\ of\ appropriate\ recommendations}{\#\ of\ possibly\ relevant\ developers}$$

Filtering Data

- Two filters
  - Exclude reports which are marked as
    - WONTFIX
    - Duplicate
    - INVALID
    - WORKSFORME
  - Excludes reports for which the developer profiles do not meet the following
    - At least 3 bugs per month

Results:
Different machine learning Techniques

Predictions	Naïve Bayes		SVM		C4.5	
	Firefox	Eclipse	Firefox	Eclipse	Firefox	Eclipse
1	59/2	54/6	64/2	58/7	64/2	40/5
2	59/2	49/11	52/3	52/13	41/3	34/9
3	59/2	44/15	57/6	47/16	42/5	31/12

Discussions on Result-I

- Is > 50% precision good enough?
- Why recall is too low?
- Applicability – Inadequate data may impact the result
- 10-fold cross validation
  - For the Firefox and gcc projects, 75% and 86% of the developers have less than 100 reports with which to train a classifier.
  - Holding back of a tenth of these reports for use as a test set represents a substantial loss of information for those developers

Validation using gcc

- Precision and recall are too low
  - Precision about 6% for 1 recommendation, 18% for 2 or 3 recommendations
- Possible reasons
  - Project characteristics
    - One developer dominates the bug fix activity
    - Labeling heuristic may not be accurate
    - Spread of bug resolution activity is too low
- Data for measuring precision and recall may not be accurate and adequate

Extensions/Alternatives

- Unsupervised learning
- Incremental machine learning
  - Precision only 28% with incremental Naïve-Bayes
- Additional data
  - Exception traces are not common for all reports
  - Also they are misleading and may have degrade the accuracy

Results-II:
Component based classification

Predictions	Eclipse	Firefox	gcc
1	86/12	64/2	6/0.4
2	82/23	64/5	10/1
3	77/32	53/6	10/2

Discussion on Results-II:

- Improvement observed for only Eclipse project

	Project Components	Min	Mean	Max
Eclipse	17	1	1.8	9
Firefox	30	1	2.9	29
gcc	30	1	3.2	28

Challenges and Lessons Learned

- Need to correlate data between CVS and Bugzilla – not always trivial
- For Mozilla Firefox committer is not necessarily the person who fix the bug (CVS log cannot be used)
- Inconsistent inclusion of references to bug report ids in the CVS comments
- During data collection, computer got blacklisted by Bugzilla server firewall

Summary

- Presented an approach to semi-automate the assignment of a bug report to a developer with the appropriate expertise to resolve bug report
- For the Eclipse and Firefox projects, achieved precision rates of over 50%, reaching 64% on one recommendation for Firefox
- The results for the gcc project were far worse, only 6% because of characteristics of the project
- Reported on lessons learned in trying to make use of data in the bug repository.
- The approach shows promise for improving the bug assignment problem for open source software developments.

Discussion

Points to Ponder

1. How practical is this methodology to be included in the Bug triage process?
2. Can there be a link between software version control and bug tracking system?
3. Can accuracy be improved?
4. How valid are the heuristics/assumptions?