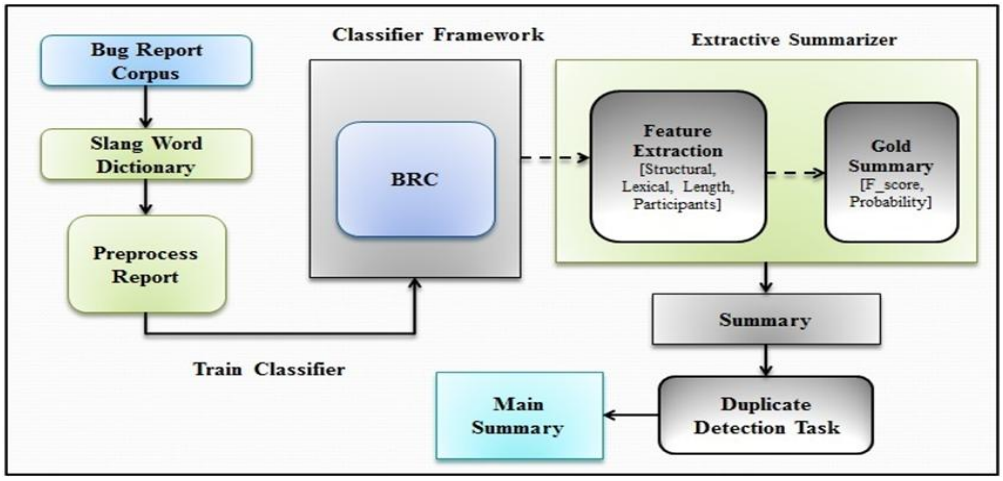

Figure 1 : System Architecture

3.3 Preprocess

In preprocessing phase of summarization, we break the text document into sentences, sentences are further broken into words and after that stop words are removed. Preprocessing phase involves four steps:

- Segmentation :
  In segmentation phase, sentences are segmented based upon sentence boundary. On every sentence boundary, the sentence are broken and put into list of lists. The output of sentence segmentation phase is collection of sentences that are further processed in next phases.

- Tokenization :
  Tokenization is the process of braking down the sentences into words.

- Stop Words Removal :
  Most commonly or frequently used words are called stop words. Stop words are meaningless and does not have any importance into the sentences. So these types of words should be removed from input document, otherwise the sentence containing more no of stop words could have higher weight.

- Root Word Identification :
  Root word identification is the process of identifying and converging words towards their root (stem). In most of the cases, variants of words having similar meaning when we interpret them.

3.4. Classifier Framework

The bug report corpus is input for producing bug report summaries automatically. Using binary classifiers that consider 24 sentence features so the proposed approach produce summaries for bug reports by using this classifiers. Based on values of these features, computed for each sentence, that it is determined whether the sentence should be involved in the summary. Manually generated summaries are time consuming but to get new feature it will be useful. So to assign a weight to each feature, a classifier first has to be trained on human generated summaries.

So here the classifier is considered which is trained on human generated summaries as follows:

- The BRC classifier, using the already created bug report corpus. To form the training set for BRC, combined the three human annotations for each bug report by scoring each sentence of a report based on the number of times it has been linked by annotators.

For each sentence, the score is between zero, when it has not been linked by any annotators, when all that annotators have a link to the sentence in their abstractive summary. A sentence is considered to be part of the extractive summary if it has a score of two or more.

3.5. Extractive Summarizer

An extraction technique of bug report summarization consists of selecting important sentences from source document(bug report) and arrange them in the destination document. Our main focus is on extraction technique for bug report summarization. Usually, the information in a given document is not constant, which means that some parts of document are more important than others are less important. The main challenge is to identify important parts of document and extract them for final summary. Here most work presented on single-document summarization using extraction method.

Processing

Processing phase is the heart of summarization; here detailed analysis on text document is done. In processing phase, feature value for every sentence is calculated. In summarization some old features and some new features are used for calculating sentence score are shown below:-

The 24 features can be categorized into four major groups.

1. Structural features are related to the conversational structure of the bug reports. Examples include the position of the sentence in the comment and the position of the sentence in the bug report.

2. Sentence Location: Location of sentence tells its importance in a text document. Starting sentences are important in almost all the cases because they express theme of the document and has higher probability to be extracted for the summary. Sentence location value is calculated in such a way that, higher values are assigned to the starting sentences and lower values are assigned to ending sentences.

3. Participant features are directly related to the conversation participants. For example if the sentence is made by the same person who filed the bug report.

4. Length features include the length of the sentence normalized by the length of the longest sentence in the comment and also normalized by the length of the longest sentence in the bug report.

5. Sentence Length : Sentences which are shorter in length may not represent theme of a text document because of fewer words contained in it, although selecting longer length sentences are also not good for summary. So sentence length values are calculated in such a way that, shorter and longer sentences are assigned lower values.

6. Lexical features are related to the occurrence of unique words in the sentence.

4. Analytic Evaluation

Recall: It evaluates proportion of relevance included in the summary.

$$R = \frac{\text{Retrieved Sentences} \cap \text{Relevant Sentences}}{\text{Relevant Sentences}}$$

Precision: It evaluates correctness for the sentences in the summary.

$$P = \frac{\text{Retrieved Sentences} \cap \text{Relevant Sentences}}{\text{Retrieved Sentences}}$$

Where, Retrieved Sentences are retrieved from the system and Relevant Sentences are identified by human.

F-score: F-score combines the values of two other evaluation measures: precision and recall. As there is always a trade-off between precision and recall, the F-score is used as an overall measure

$$\textbf{F-score} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

5. Duplicate Detection Task

In this task duplicate sentences from bug reports are removed by calculating sentence value and sentence similarity. The following two approaches are used to describe the duplicate detection task. .

1. Lexicon based approach(TF-IDF)

Lexicon based approach is formed by calculating tf–idf of summary, tf means term frequency and idf means inverse document frequency, is a numerical evaluation of that how important a word is to a document in a corpus. This approach is using as a weighting factor in information retrieval.

The tf-idf calculates possibility value of the number of times a word appears in the document. Variations of the tf–idf weighting scheme are often used ranking a document's relevance given a user query. It can be effectively used for stop-words sifting in various subject fields including text summarization and classification.

2. Concept based approach:

Concept based approach is introduce a word synonyms from the sentence. It compares sentences from the summaries and explores the similar sentences and removes that sentence from summary. And used word net 3.0.

These both approach refers Jaccord's coefficient of similarity for duplicate detection task.

6. Results

If considering comparison of classifiers from previous approach there is no significant difference when comparing the performance of EC and EMC so the results obtained for the EC and EMC classifiers were similar to those produced when the same classifiers applied to meeting and email data.

The results demonstrated that based on standard measures, while classifiers trained on other conversation-based data (EC and EMC) generated reasonably good bug report summaries and a classifier specifically trained on bug report data (BRC) also generated summaries that are better with statistical significance.

So the proposed system is based only on bug report summary generation. Bug report summaries are intended to help a subject save time performing a bug report duplicate detection task by not having to interact with bug reports in their original format. At the same time it is expected that summaries contain enough information so that the accuracy of duplicate detection is not compromised.

1. To producing accurate results for bug repositories the proposed system goal is to develop a summarization approach. So dataset means bug repository contains bug reports here using KDE, Mozilla, Red hat open source projects bug repositories. The bug reports contains conversational content and avoided selecting bug reports consisting long stack traces and large chunks of code, so bug reports are with mainly natural language content. Preprocessing phase is a training phase which trains the classifier using slang words dictionary.

2. After preprocessing phase getting summarized report.

3. By removing duplicate bug reports using the method post-

processing technique. After that finally get summary of bug reports.

4. Summary Report in PDF format with evaluation result.

Title: crash in QMetaObject activate while accesing samba share
Status: NEEDSINFO
Bug Reported By: Miikka Salminen
Description:
Decreasing the number of shown WLAN networks can cause a stale, empty entry (hovering the mouse on the entry doesn't highlight it) with zero signal strength appear at the top of the list with a following empty space between the entry in question and the second, areeal network.These entries take youp space and push the areeal entries downwards, which causes them to overlap with the VPN connections.

----------Summary----------

Resolved in 0.8 Bug 188320 has been marked as a duplicate of this bug . When disabling the WLAN from the checkbox in the list , a stale , empty , younhighlightable entry with zero signal strength is left on the list . Same areoot cause , so also fixed This bug has been marked as a duplicate of bug 188319 . Alot of smb : \/ kio crashes were fixed on KDE4 .2 . xso I grinuess this should be fixed too . Please test with KDE4 .2 . xonce you can install it .

Total No.Of Sentences in Bug Report: 11
Total No.Of Sentences in Summary: 7
Summary Size: 0.6363636363636364

7. Conclusions

Using automatically generated software artifact like bug reports are used to provide developers multiple benefits and existing conversation-based extractive summary generators can produce summaries for reports that are better than a random classifier. An extractive summary generator trained on bug reports produces the best results. Generated bug report summaries could help developers perform duplicate detection tasks in less time with no indication of accuracy degradation, confirming that bug report summaries help software developers in performing software tasks.

8. Acknowledgement

I would like to thank my project guide Prof. S. A. Shinde sir for giving his valuable guidance, inspiration and encouragement to embark this paper. Prof. S. A. Shinde sir gave me all the freedom I needed for this project. This project being conceptual one needed a lot of support from my guide so that I could achieve what I was set out to get.

References

[1] Sarah Rastkar, Gail C. Murphy and Gabriel Murray, "Automatic Summarization of Bug Reports,"IEEE Transactions on Software Engineering,2013.
[2] Nenkova and K. McKeown, "Automatic summarization," Foundations and Trends in Information Retrieval, vol. 5, no. 2-3, pp. 103–233, 2011.
[3] G. Murray and G. Carenini, "Summarizing spoken and written conversations," in EMNLP'08: Proc. of the 2008 Conference on Empirical Methods on Natural Language Processing, 2008.
[4] J. Anvik, L. Hiew, and G. C. Murphy, "Coping with an open bug repository," in Proc. of the 2005 OOPSLA Workshop on Eclipse Technology eXchange, 2005, pp. 35–39.
[5] J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?" in ICSE'06: Proc. of the 28th International Conference on Software Engineering, 2006, pp. 361–370.
[6] J. Davidson, N. Mohan, and C. Jensen, "Coping with duplicate bug reports in free/open source software projects," in VL/HCC'11: Proc. of the 2011 IEEE Symposium on Visual Languages and Human-Centric Computing, 2011, pp. 101 108.
[7] O. Rambow, L. Shrestha, J. Chen, and C. Lauridsen, "Summarizing email threads," in HLT-NAACL'04: Proc. of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, 2004.
[8] 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 ICSE'08: Proc. of the 30th International Conference on Software Engineering, 2008, pp. 461–470
[9] P. Runeson, M. Alexandersson, and O. Nyholm, "Detection of duplicate defect reports using natural language processing," in ICSE'07: Proc. Of the 29th International Conference on Software Engineering, 2007, pp. 499–510

Author Profile

Miss. Rutuja Taware received the B.E. degrees in Information Technology from Pune University in 2013 and 1999, and pursuing M.E. degree from VPCOE, Baramati. She has attended various workshops organized by IITB and IITD with remote center. She has attended "Computer vision" by Dr. Sumantra Dutta Roy. She attended workshops of Latex and SciLab conducted by remote center of VPCOE under IITB. He has published Review paper on same topic in IJERGS. She has participated in cPGCON-2015 i.e. post Graduate Conference.

Prof. Santosh Shinde received his B.E. degree in computer engineering (First Class with Distinction) in the year 2003 from Pune University and M. E. Degree (First Class with Distinction) in Computer Engineering in 2010 from Pune University. He has eleven years of teaching experience at undergraduate and postgraduate level. He has attended various National and International Conferences, Seminars and Workshops on various subjects like Multimedia Techniques, Multicore Computing and Software Engineering. He is a life member of ISTE (Indian Society for Technical Education) and IACSIT (International Association of Computer Science and Information Technology). He has worked as a review committee member for the 1st International Conference on recent trends in Engineering and Technology (ICRTET'2012) held at SNJB's COE, Nashik. He has worked as a Judge for the State level paper presentation (REBEL) held at SVPM's COE, Malegaon. He has attended workshops on Robotics, Research paper writing and Latex conducted by IIT, Bombay remote center at VPCOE, Baramati. He has to his credit IBM RFT certification with 100% score. He has organized one day workshop on Software Engineering by Dr. S. A. Kelkar, adjunct Professor IIT, Powai.