MACHINE INTELLIGENCE RESEARCH

ISSN:2731-538X | E-ISSN:2731-5398
Vol. 18 No. 01 (2024)

NEW ASSESSMENT PREDICTION MODEL OF ECLIPSE REPORTS BASED STATISTICAL ALGORITHMS

[1]Bikka Venkata Pranay Kumar Reddy [2]Dr. Y. Chitti Babu [3]Ramesh Kunchala
[1]M. Tech Scholar, Dept.of CSE, St. Ann's College of Engineering & Technology, Chirala.
[2]Associate Professor, Dept.of CSE, St. Ann's College of Engineering & Technology, Chirala.
[3]Assistant Professor, Dept.of CSE, St. Ann's College of Engineering & Technology, Chirala.
e-mail: chitti510@gmail.com

ABSTRACT: The rapid growth of software scale and complexity, a large number of bug reports are submitted to the bug tracking system. In order to speed up defect repair, these reports need to be accurately classified so that they can be sent to the appropriate developers. Software fault prediction is a vital and helpful technique for boosting the quality and dependability of software. There exists the prospective to enhance project management by proactively estimating prospective release delays and implementing cost-effective measures to boost software quality. The subsets of queries extracted and then each model was analyzed how it deals with specific group of queries. The aim is to build a tool that automatically classifies software bugs according to the severity and priority of the bugs and makes predictions based on the most representative features and bug report text. We present a machine learning based solution for the bug assignment problem. We build component classifiers using a multi-layer Neural Network based on features that were learned from data directly. A hierarchical classification framework is proposed to address the mixed label problem and improve the prediction accuracy. The features are used K-Nearest Neighbor, Naive Bayes, Logistic Regression, Support Vector Machine and Random Forest) show that our proposed method achieves better performance The implementation of this study makes use of methods from AI, along with data mining, and Machine Learning, along with statistical algorithms, and also modeling. Prediction models can be of assistance in maximizing all of the resources needed for the research.

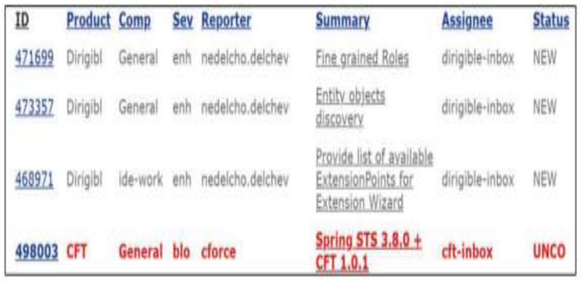
INDEX TERMS Natural Language Processing, Machine comprehension, Deep learnin. Machine Learning, ML, Software Bug, Bug Priority, Bug Detection, Software Security.

1. INTRODUCTION

As more and more features and functionalities are added to a software system it is inevitable that software bugs will emerge. To fix them timely, bugs have to be assigned to the right developers [1]. A software bug is a failure in the program which causes unexpected or unwanted outputs is an error that prevents the program to operate its function as it should either while launching the software using its features system operators and software developers spend huge time testing their proposed software as modules to bypass having any type of bugs and assessing the potentials of having any type of system crashes for any reason [2]. In presented work an answer comparison mechanism has been defined and implemented to obtain a final answer based on separated answers given by chosen models comparative studies were performed between models with particular reference to their attention layers and analysis of the results gained by models, including error analysis [3]. Early prediction and detection of problematic parts should typically

ISSN:2731-538X | E-ISSN:2731-5398
Vol. 18 No. 01 (2024)

demand quick debugging the nature of which is determined by the severity degree of the defect or defects that have been found [4]. In addition the phase of the software development process known as the gathering of software requirements is an important early stage [5]. Machine learning algorithms take the input as a series of feature vectors which means that one has to produce these numerical values for each entry [6]. Hence, researchers tend to reuse existing datasets in order to reduce the amount of work to be done and increase the reproducibility of their approaches [7]. These wrong tags will cause the bug report to not be correctly assigned to the appropriate developers thereby increasing the difficulty of defect repair [8]. In order to reduce this impact and accelerate the speed of defect repair the software engineering industry needs accurate and automated classification methods for bug reports [9].



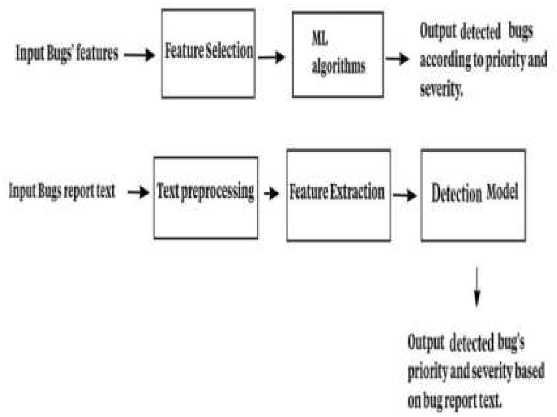
ID	Product	Comp	Sev	Reporter	Summary	Assignee	Status
471699	Dirigibl	General	enh	nedelcho.delchev	Fine grained Roles	dirigible-inbox	NEW
473357	Dirigibl	General	enh	nedelcho.delchev	Entity objects discovery	dirigible-inbox	NEW
468971	Dirigibl	ide-work	enh	nedelcho.delchev	Provide list of available ExtensionPoints for Extension Wizard	dirigible-inbox	NEW
498003	CFT	General	blo	cforce	Spring STS 3.8.0 + CFT 1.0.1	cft-inbox	UNCO

Figure.1. Several examples of bug report from Bugzilla

2. RELATED WORKS

Bug report classification helps developers understand and fix software defects to the skyrocketing number of bug reports, manual classification has become time-consuming and laborious for a long time researchers have been exploring how to implement automatic classification of bug reports [10]. NN can effectively fit random nonlinear data and features self-learning capability after proper parameter [11].The most widely used machine learning algorithms for bug prediction are Logistic Regression Naive Bayes, Decision Tree, and Random Forest ensemble learning techniques have started to be adopted in the context of bug prediction [12]. Additionally they proposed an analytical model so as to accurately evaluate the efficiency of remediation techniques for various fault types. Related to a reactive fault controlling system the results of an experiment using a specifically designed prototype demonstrate enhanced availability with reduced overhead [13]. According to Ahmed the framework is created utilizing NLP along with supervised machine learning methods. It allows for modeling the vector representation of the context paragraph at different structural levels: character level, word-level and contextual-level. The architecture is based on the bidirectional attention flow mechanism [14]. To avoid information loss caused by early summarization, attention is computed at each time step, instead of summarizing the context into one fixed-size vector [15]. The imbalanced data problem can be solved using the oversampling methods, such as SMOTE. Bugs' reports have a summary or description field as text at the bug prediction process there is a need to deal with this text and convert it to numeric format to apply the machine learning algorithms [16].

ISSN:2731-538X | E-ISSN:2731-5398
Vol. 18 No. 01 (2024)


Input Bugs' features
Feature Selection
ML algorithms
Output detected bugs according to priority and severity.
Input Bugs report text
Text preprocessing
Feature Extraction
Detection Model
Output detected bug's priority and severity based on bug report text.
Figure 2. Bugs classification and prediction model architecture

3. SYSTEM ARCHITECTURE

We collect and manually mark bug reports in the open repository and then perform preprocessing steps on them. We use BERT and TF-IDF methods to extract features. And the text feature and frequency feature are merged and normalized. We input the extracted features into five classifiers [17].The preprocessing of training data starts with the deletion of unnecessary class and file-related information for every entry such as filename, parent, path. Furthermore we binaries the target labels, converting the number of bugs found in a class to 0 or 1 [18]. Introduces two novel concepts to approaching the reading comprehension task First authors present a re-attention mechanism Second they show a dynamic-critical reinforced learning approach to training models. The BERT model is a pre-training model proposed by Google which can learn dynamic context word vectors and more comprehensively capture the features of word meaning, word position and sentence meaning [19]. In this experiment the output of the penultimate layer of the BERT model is used as the feature score.

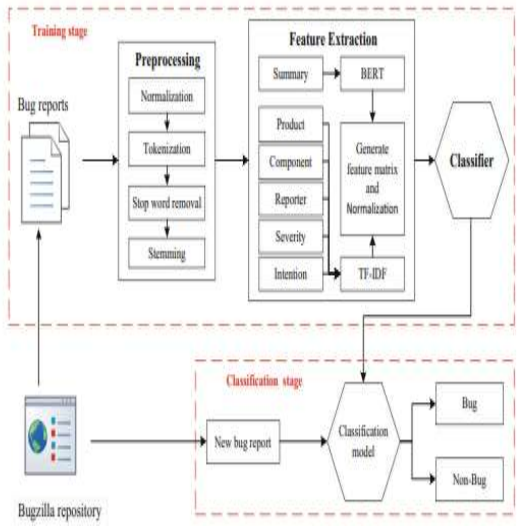


Figure.3. Framework of our approach

4. PROPOSED SYSTEM

ISSN:2731-538X | E-ISSN:2731-5398
Vol. 18 No. 01 (2024)

In case most software programs have become large and complicated, software bugs need to take into consideration. Detecting and fixing bugs problem at initial levels that reflected positively on the quality, security, and performance of the program and will save time and effort. Machine learning algorithms and Natural language processing techniques (NLP) have a great effort in software bugs classification and prediction [20]. The statistical analysis of the SQuAD questions as well as an analysis of the results obtained standalone architectures with regard to the question classes. It shows how the SQuAD data set has been splatted in order to perform experiments [21].`

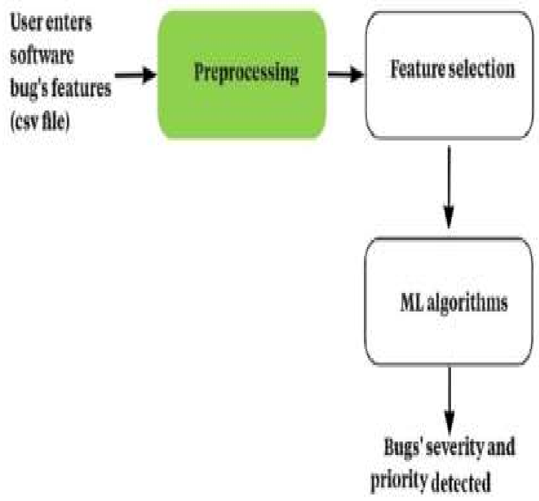
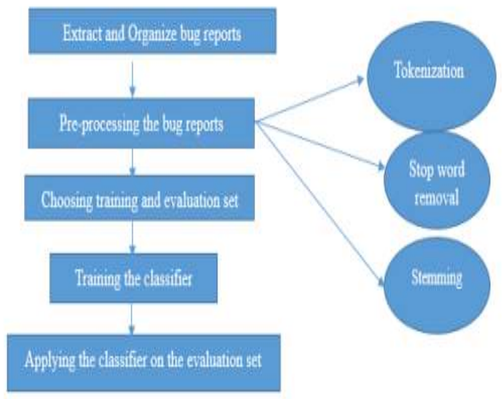

User enters software bug's features (csv file)
Preprocessing
Feature selection
ML algorithms
Bugs' severity and priority detected

Figure 4. Our proposed Bugs' severity and priority classification and prediction system

5. METHODOLOGIES

One cannot completely eradicate bugs, fixes, patches, and other software-related issues because each one has a severity and priority level dataset utilized in this study, focuses on the predictive analysis of software development frameworks in relation to software bug attributes severity, and priority the dataset for agile software development was created [22]. To overcome this issue we used two techniques: feature selection and oversampling (the Synthetic Minority Oversampling Technique (SMOTE)) along with ensemble learning method. The evaluation metrics used are the level of accuracy, F1 score, precision, and recall repeated in the corpus that may help to distinguish between the types of bugs' severity [23].


Extract and Organize bug reports
Pre-processing the bug reports
Choosing training and evaluation set
Training the classifier
Applying the classifier on the evaluation set
Tokenization
Stop word removal
Stemming

ISSN:2731-538X | E-ISSN:2731-5398
Vol. 18 No. 01 (2024)

Figure 5. Proposed Methodology of this study

6. GENETIC ALGORITHMS

Random includes two aspects: one is for the training process. In order to ensure that all samples have a chance to be drawn once, the classifier randomly selects a training sample set and the data used in each round of training is randomly selected from the original sample set with replacement [24].

Algorithm Weighted class-specific voting ensemble Training data: full SQuAD training data set Evaluation data: full SQuAD evaluation data set

Step1: Each model is trained on the Training data.
Step2: For each evaluation question, a question class is identified.
Step3: Each trained base model is asked to answer the questions from the Evaluation data, resulting in three candidate answers per question.
Step4: Each candidate answer gets a weight that corresponds to the question class and the answering model.
Step5: if class is different from undefined then
Step6: if one candidate answer is a duplicate of another one then
Step7: Their weights are added together
Step8: The ensemble returns the candidate answer that has the highest weight.
Step9: else
Step10: The candidate answer produced by the globally best model is returned
Step11: else
Step12: The candidate answer produced by the globally best model is returned

7. EXPERIMENT RESULTS

This experiment divides the training data and test data by 8:2, and extracts the features of the report summary field, other fields, and intention, respectively. In order to find the most suitable classifier for the proposed method In order to improve the result and the model performance, two techniques were used, Feature selection and Oversampling techniques. SMOTE focuses on samples near the border of the optimal decision function and will generate samples in the opposite direction of the nearest neighbors' class and connect inliers and outliers. The models encounter challenges in accurately forecasting classes other than 5 and 1 in terms of severity and importance. However, it is in these specific classes that the models demonstrate exceptional performance.

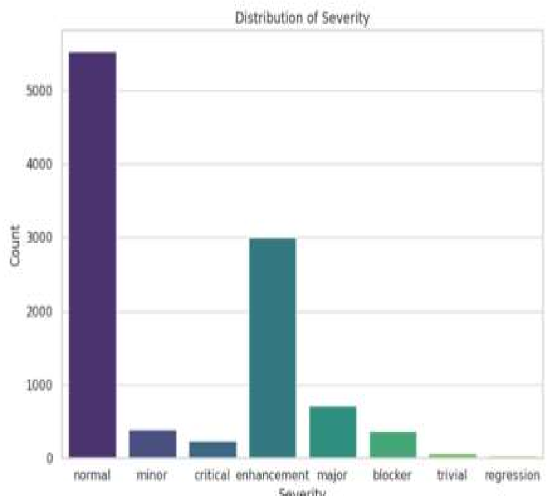


Figure 6. Distribution of Severity

8. CONCLUSION AND FUTURE WORK

We propose a new automatic classification approach for bug reports increase the intention of the report based on the text information of the report. Our approach combines Text Mining, Natural Language Processing and Machine Learning technologies. We first collected reports from the four ecosystems in the bug repository, and manually marked their types and intention, with the goal of constructing the data set required. The deeper analysis of input data was required to achieve better results than best deep learning model is performance used instead of using the bug's summary. The comparison process was done depending on Accuracy and ROC curve final results, before and after applying the over-sampling and feature-selection techniques. Out future plans include trying different ensemble learner methods combined with other base learner techniques. We would also like to try other preprocessing techniques as well in our future work. Our goal is to provide an AI assistant for people who handle bug assignment, so they can save the time of going through a long list of possible components. Based on our experiments and user feedbacks, the service works fairly reliably for this purpose.

9. REFERENCES

[1] M. A. Jamil, M. Arif, N. S. A. Abubakar, and A. Ahmad, ‘‘Software testing techniques: A literature review,’’ in Proc. 6th Int. Conf. Inf. Commun. Technol. Muslim World (ICT4M), Nov. 2016, pp. 177–182.
[2] Shatnawi, M. Q., &Alazzam, B. (2022). An Assessment of Eclipse Bugs' Priority and Severity Prediction Using Machine Learning. International Journal of Communication Networks and Information Security, 14(1), 62-69.
[3] Akmel, F., Birihanu, E., &Siraj, B. (2017). A literature review study of software defect prediction using machine learning techniques. Int. J. Emerg. Res. Manag. Technol, 6(6), 300-306.
[4] Ali, S., Ullah, N., Abrar, M. F., Majeed, M. F., Umar, M. A., & Huang, J. (2019). Barriers to software outsourcing partnership formation: an exploratory analysis. IEEE Access, 7, 164556-

ISSN:2731-538X | E-ISSN:2731-5398
Vol. 18 No. 01 (2024)

164594
[5] Brill, Eric and Dumais, Susan and Banko, Michele. An Analysis of the AskMSR Question-answering System. Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - vol. 10, pp. 257-264, 2002, Association for Computational Linguistics, Stroudsburg, PA, USA
[6] Sarkar, Sanglap and Madasu, Venkateshwar Rao and SM, Baala Mithra and Rao, Subrahmanya VRK. NLP Algorithm Based Question and Answering System. Proceedings of the 2015 Seventh International Conference on Computational Intelligence, Modelling and Simulation, CIMSIM '15, 2015, pp. 97-101, IEEE Computer Society, Washington, DC, USA
[7] K. R. K. V. Prasad , V. Srinivasa Rao, P. Harini , Ratna Raju Mukiri , K. Ravindra, D. Vijaya Kumar , and Ramachandran Kasirajan "Machine Learning Algorithms Are Applied in Nanomaterial Properties for Nanosecurity". Hindawi Journal of Nanomaterials Volume 2022, Article ID 5450826, 14 pages https://doi.org/10.1155/2022/5450826
[8] Wenpeng Yin and Katharina Kann and Mo Yu and HinrichSchütze. Comparative Study of CNN and RNN for Natural Language Processing. CoRR, vol. abs/1702.01923, http://arxiv.org/abs/1702.01923, 2017.
[9] Hochreiter, Sepp and Schmidhuber, Jürgen. Long Short-Term Memory. Neural Computing, vol.9, pp.1735-1780, November 15, 1997, MIT Press, Cambridge, MA, USA
[10] Sharma, Yashvardhan and Gupta, Sahil. Deep Learning Approaches for Question Answering System. vol.132, pp.785-794, Procedia Computer Science, 2018
[11] Tomas Mikolov and Kai Chen and Greg Corrado and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. CoRR, vol.abs/1301.3781, http://arxiv.org/abs/1301.3781, 2013
[12] Jason Weston and Antoine Bordes and Sumit Chopra and Tomas Mikolov. Tow ards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. vol.abs/1502.05698, http://arxiv.org/abs/1502.05698, 2015
[13] Jason Weston and Sumit Chopra and Antoine Bordes. Memory Networks. CoRR, vol.abs/1410.3916, http://arxiv.org/abs/1410.3916, 2014
[14] Ankit Kumar and OzanIrsoy and Jonathan Su and James Bradbury and Robert English and Brian Pierce and Peter Ondruska and Ishaan Gulrajani and Richard Socher. Ask Me Anything: Dynamic Memory Networks for Natural Language Processing. CoRR, vol.abs/1506.07285, http://arxiv.org/abs/1506.07285, 2015
[15] Minh-Thang Luong and Hieu Pham and Christopher D. Manning. Effective Approaches to Attention -based Neural Machine Translation, CoRR, vol. abs/1508.04025, http://arxiv.org/abs/1508.04025, 2015
[16] Akimova, Elena N., Alexander Y. Bersenev, Artem A. Deikov, Konstantin S. Kobylkin, Anton V. Konygin, Ilya P. Mezentsev, and Vladimir E. Misilov. 2021. "A Survey on Software Defect Prediction Using Deep Learning" Mathematics 9, no. 11: 1180. https://doi.org/10.3390/math9111180
[17] Rudolf Ferenc, DénesBán, TamásGrósz, Tibor Gyimóthy, Deep learning in static, metric-based bug prediction, Array, Volume 6, 2020, 100021, https://doi.org/10.1016/j.array.2020.100021.

ISSN:2731-538X | E-ISSN:2731-5398
Vol. 18 No. 01 (2024)

[18] S. Iqbal, R. Naseem, S. Jan, S. Alshmrany, M. Yasar and A. Ali, "Determining Bug Prioritization Using Feature Reduction and Clustering With Classification," in IEEE Access, vol. 8, pp. 215661-215678, 2020,
[19] Yang, Zichao and Yang, Diyi and Dyer, Chris and He, Xiaodong and Smola, Alex and Hovy, Eduard. Hierarchical Attention Networks for Document Classification. Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1480-1489, 2016, San Diego, California
[20] Nikolaos Pappas and Andrei Popescu-Belis. Multilingual Hierarchical Attention Networks for Document Classifi-cation. CoRR, vol. abs/1707.00896, http://arxiv.org/abs/1707.00896, 2017
[21] SainbayarSukhbaatar and Arthur Szlam and Jason Weston and Rob Fergus. Weakly Supervised Memory Networks. CoRR, vol. abs/1503.08895, http://arxiv.org/abs/1503.08895, 2015
[22] Jeffrey Pennington and Richard Socher and Christopher D. Manning. booktitle = Empirical Methods in Nat-ural Language Processing (EMNLP), GloVe: Global Vectors for Word Representation, pp. 1532-1543, 2014, http://www.aclweb.org/anthology/D14-1162
[23] Adams Wei Yu and David Dohan and Minh-Thang Luong and Rui Zhao and Kai Chen and Mohammad Norouzi and Quoc V. Le, QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. CoRR, vol. abs/1804.09541, 2018.
[24] Minghao Hu and Yuxing Peng and XipengQiu. Mnemonic Reader for Machine Comprehension. CoRR, vol. abs/1705.02798, http://arxiv.org/abs/1705.02798, 2017
[25] Hsin-Yuan Huang and Chenguang Zhu and Yelong Shen and Weizhu Chen. FusionNet: Fusing via Fully-Aware At-tention with Application to Machine Comprehension. CoRR, vol. abs/1711.07341, http://arxiv.org/abs/1711.07341, 2017