Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 10, Issue. 12, December 2021, pg.21 – 26

Bug Reports and Deep Learning Models

Som Gupta[1]; Sanjai Kumar Gupta[2]
[1]Research Scholar Computer Science Engineering Department & AKTU Lucknow, India
[2]Associate Professor Computer Science Engineering Department & BIET Jhansi UP, India
[1] somi.11ce@gmail.com; [2] guptask_biet@rediffmail.com
DOI: 10.47760/ijcsmc.2021.v10i12.003

Abstract— Deep Learning is one of the emerging and trending research area of machine learning in various domains. The paper describes the deep learning approaches applied to the domain of Bug Reports. The paper classifies the tasks being performed for mining of Bug Reports into Bug Report Classification, Bug Localization, Bug Report Summarization and Duplicate Bug Report Detection. The paper systematically discusses about the deep learning approaches being used for the mentioned tasks, and the future directions in this field of research.

Keywords— Deep Learning, Convolutional Neural Networks, Gated Recurrent Unit, Neural Networks, Bug Reports

I. INTRODUCTION

During a software development process, lot of artifacts are produced for the better communication and for the future reference. These artifacts are generally stored in the online repositories. It has been observed that mining these artifacts help get lot of insights related to the project like from the software project management perspective, for improving the system performance and as a referral for other projects similar to them.

Bug Reports and Software Code are the two artifacts which are popularly used for mining purpose. Bug Reports are one of the main artefacts which are widely referred during the software maintenance task. When it comes to Mining Software Repositories for getting the insights into the project evolution from various perspectives, software code and the Bug Reports are the most widely mined artifacts. Bug Reports are the artifacts which not only the developers create but it is a resource which is public and can be used or can be developed by anybody using the product. There are many analysis that have been performed over the Bug Reports like summarizing the Bug Reports for quick retrieval of relevant report, localizing the bugs in the source code by connecting with the Bug Reports, Assigning the Bug Reports to the concerned developer, understanding whether the Bug Report is duplicate or not, determining the validity of Bug Report, etc.

Lot of approaches have been used to analyse the Bug Reports which involves the usage of Information Retrieval Techniques, NLP Based Approaches, Machine Learning Approaches, and Soft Computing Based Approaches. Vector Space Models, Similarity Based Approaches, LDA, TFIDF are few of the very widely used approach being used in IR Based Researches. Abstract Syntax Trees, Parsing, Word Embedding's are few used for NLP Based Approaches. Decision Trees, Naïve Bayesian, K-Nearest Neighbor, and SVM are in the Machine Learning Based Approaches. Soft computing techniques involve the use of Fuzzy Logic, Genetic Programming, Optimization techniques like Particle Swarm Optimization, etc. Word2Vec, Glove, FAST, Convolutional Neural Networks, Recurrent Neural Networks, LSTM (Long Short Term Memory) are few famous approaches being used for the Deep Learning Based Approaches. Convolutional Neural Networks consist of input layer, output layer, pooling layers, and normalization layers. They have been popularly used for the task of image recognition.

Deep Learning Models are one of the trending approach which the researchers are using for analysing the documents. Deep Learning models have been used not only for Software Artifacts but also to various domains like speech recognition, image processing, robotics, computer vision, database related researches and many others. Tensor Flow and KERAS are one of the most popular library which is used for executing the deep learning models for the various scenarios. It has been observed that in GitHub there are more than 3000 deep learning based applications which have used Tensor Flow to build up the models [1].

Objectives of the paper are as follows:

O1: How much work has been done in Bug Reports Analysis using Deep Learning Models?

O2: What are the popular Deep Learning Models used for Bug Report Analysis

O3: What are the main mining tasks on the Bug Reports where the deep learning models have been used

O4: What are the challenges using deep learning models to this domain

The paper is organized as follows: Section 1 introduces the Bug Reports and the Deep Learning Models. Section 2 classifies the Bug Report analysis tasks and states the approaches being used by various researchers utilizing the deep learning models. Section 3 mentions the challenges of the deep learning models and the reason for very less work in this field. And finally the conclusion.

II. CLASSIFICATION OF BUG REPORT TASKS

Lot of works have been performed for the analysis of Bug Reports. In this paper we have analysed the various works and we classify them to Bug Report Severity Classification, Bug Report Summarization, Duplicate Bug Report Detection, and Bug Report Triaging.

SNo.	**Classification of Analysis Task**		
	Task	**Deep Learning Approaches Used**	**Famous Researches**
1	Bug Report Classification	LSTM Variational Auto Encoders	Chauhan et al.[1] Chen et al.[2] Tan et al.[3] Madhu et al. [4].
2	Bug Report Summarization	Auto Encoder Model, Restricted Boltzmann Machine	Verma et al. [5] Li et al. [6]
3	Bug Report Triaging	Recurrent Neural Networks, LSTM	Cao et al. [7] Liu et al [8] Mani et al. [9] Florea et al. [10] He et al. [11]
4	Bug Localization	Convolutional Neural Networks	Lam et al. [12][13] Xiao et al. [14] Sharma et al. [15]
5	Duplicate Bug Report Detection	CNN and LSTM Models	Budhiraja et al. [16] Xie et al. [17] Deshmukh et al, [18] Kukkar et al. [19]

Bug Report Classification: It is one of the major activity involved during the testing and the evolution phase of software engineering. Because of the increased competitiveness, agile software methodology is the approach that most of the application developers use. Agile Approaches are based upon the less time to market and the quick revisions or upgrades to build up the application according to the changing customer's needs. After every version, the software developers give the opportunities to the users to report the bugs. Mostly the companies use open source applications like Bugzilla, JIRA, etc. to track and maintain the Bugs. Users report the Bugs in the form of Bug Reports. Classifying the Bug Reports according to their severity is one of the very important task as incorrect classification may lead to non-attending of the high severity Bugs in their application and thus ultimately leading to unnecessary delays and the non-satisfaction of the users towards their application. Manual process of Bug Report Classification is prone to the errors, time-consuming, and expensive. Thus automatic

ways have been researched to classify the Bug Reports. Most of the researches have classified the Bug Reports into Critical, Major, Moderate, Minor and cosmetic severity levels. SVM, Naïve Bayesian, Random Forests, Classification Trees are few of the major approaches being used for this approach.

Chauhan et al. [1] have used the Convolutional Neural Networks along with the L1 and L2 regularization techniques to classify the Bug Reports according to the severity. They collected the Bug Reports, performed the text pre-processing by involving the tasks like tokenization, stop word removal, stemming; then extracted the bigrams and the trigrams from the text; collected the features with the labels and then used the Convolutional Neural Networks. They used the pooling, and soft max model over it. Multiple layers were used in their model. ReLU activation function was also used. They used the Precision, Recall, F-Score, and Accuracy as the metrics to evaluate the model.

Chauhan et al. [2] where they classified the Bug Reports according to the severity; they classified into environment configuration, input data based bugs, version specific, and performance specific

Chen et al. [3] in his work observed how the performance related Bug Reports were impacted by the use of deep learning models..

Tan et al. [4] in his thesis classified the Bug Reports as security-related and non-security-related. For this task they used the auto encoders, Recurrent Neural Networks, LSTM Based Models. In their work, they first pre-processed the text into title, subject and description. Then they used the supervised models, unsupervised models and the deep learning models to perform the analysis. For the supervised approaches, they have used Naïve Bayesian along with 10 fold cross validation. For unsupervised learning, they used the Principal Component Analysis (PCA) and Variational Auto Encoders (VAE). For the deep learning models, they have used the LSTM Based models. The purpose of using LSTM Models is that they store the states of the model and helps decide what has to be stored and what has to be forgotten. The data was fed as in the form of word embedding. ReLU was used for optimization along with the sigmoid functions.

Madhu et al [5] have used the deep learning models along with the entropy measures to prioritise the Bug Reports.

Bug Report Summarization:

Bug Reports are one of the major artifacts which are involved during the regression testing of the application. Many times when a similar Bug appears, the developers look into the bug repositories to find the resolution process of the similar bug report. Reading and going through each and every bug report is a tedious task and thus summaries help in locating the right bug report quickly. Automatic summarization is one of the step which helps in easy localization and increases the comprehension of the Bug Report. The summarization techniques have been widely classified into extractive and the abstractive summaries according to whether the summaries are generated by extracting the important sentences and arranging them or extracting the important sentences, rephrasing them and then generating the novel sentences.

As the extractive summaries are easy to generate and are not very expensive, these are the widely used for the summarization purpose. Many models have been used for generating these summaries like Naïve Bayesian, Hidden Markov Models, SVM, and Rank Based Approaches. But now after observing the strengths and the performance of deep learning models in other domains, researches are analysing their performances in the field of summarization also.

Verma et al. [6] used the feature extraction, feature enhancement and the summary generation phases by involving the use of Neural Networks like Restricted Boltzmann Machine with 1 hidden layer and 1 visible layer. Persistent Contrastive Divergence was used for sampling. Sentences were modelled to the feature vectors and were feed to the model for training. Precision, Recall and F-Measure was used for the evaluation purpose.

Li et al. [7] used the stepped auto encoder network model to generate the unsupervised bug report summaries. Their model used three hidden layers. RMSProp Optimizer was used for optimizing the network parameters. Sigmoid activation function was used for the model. The approach helps improve the vectors generated for the sentences and detect the duplicate sentences and the evaluation sentences of the Bug Report. They used the dynamic programming along the deep learning model to generate the sentences for the summary.

Bug Localization

It is the process of identifying the buggy files based upon the Bug Reports. Many approaches have been used for the Bug Localization process like Information Retrieval and Deep Learning Based Models. Variations of Vector Space Models (VSMs), Latent Semantic Indexing (LSI), Latent Dirichlet Allocation (LDA) have been used in IR community to perform the task. This is mainly done by extracting the features from the Bug Reports and the source code. Because of the demand of high dimensional data to perform the task, deep learning models are also very popular. Convolutional Neural Networks and their variations like Dynamic Convolutional Neural Networks are popular among the deep learning community for this task.

Liang et al. [8] have created one approach by using the deep learning models and the abstract syntax trees of programs. Lexical semantics of Bug Reports were also extracted to help perform this task. They used tree based

convolutional neural networks. Abstract Syntax trees helped in finding the semantically similar entities. In order to deal with unbalanced dataset, they have used the optimization functions based upon the Adam Algorithm.

Bug Report Triaging

When a Bug Report comes to the Bug Repositories, the Bug Report is sent to one developer to fix it. The process of assigning the Bug Report to a developer to resolve it is known as Bug Report Triaging. Manual Bug Report Triaging is a time consuming task and thus automated process for the same is required. In the research community, many researchers consider this task as a classification problem. They use the title and the description of the Bug Report as the input features and identifying the developer as the class label. Even though the problem appears simple, the presence of unstructured data, code and the stack traces pose the challenges to the process. LDA, BM25F, Hidden Markov Models, Naïve Bayesian, C4.5 and Support Vector Machines are the most common approaches which have been used for this purpose.

Mani et al. [9] have used Bidirectional Recurrent Neural Network (RNN) based model to perform this task. They found that the model learnt the semantic and the syntactic features from the long text. Attention mechanism was used along with the RNN model to capture the context of the text. They compared their model with the machine learning based models like Bag of words model, support vector machines, Naïve Bayesian Model, Similarity Measures and found that their model worked better than these individual models. LSTM model was used in the hidden layer. Their approach involved the collection of Bug Reports, removing the URLs, stack traces, and pre-processing of the text; constructing the vocabulary of words, and then applying the deep learning models. The sentences were represented by word2Vec.

Li et al. [2] used the back propagation techniques and multiple LDAs to improve the Bug Triaging process.

Florea et al. [10] used the Recurrent Neural Networks and the Convolutional Neural Networks for recommending the developer for Bug Report Assignment.

Valid Bug Report Detection:

Very less work has been done in detecting whether the Bug Report is valid or not. Most of the researchers use the features and used the Machine learning models like random forest, and support vector machine or the Information Retrieval Models like LDA to find the validity of the Bug Reports.

He et al. [11] used the summary and description of the Bug Report to examine if the bug report is valid or not. For performing the task, they used the CNN model to find the contextual and the semantic information from the text. Apart from just determining the validity of the Bug Report, they also categorized them into "Attachment", "Reproduce" and "Environment". Their CNN model used two matrices one for the matrix of summary and one for the matrix of description. Two independent CNNs were created for this purpose. Filters were used to extract the features of the input. Max Pooling was used with CNN to reduce the number of features. The result from pooling layer was fed to the fully connected layer which used the sigmoid and the linear classifiers to produce the output.

Duplicate Bug Report Detection

Duplicate Bug Report Detection is the process of identifying whether the current Bug Report is the duplicate of existing Bug Report. This is one of the important task in order to avoid the redundancies. This helps reduce the time and the cost. Not just they help eliminate the duplicate data but also sometimes identifying the similar Bug Report helps get more information about the Bug Report. It has been observed that on daily basis more than thousands of Bug Reports are detected as duplicates. Even though lot of works have been performed in this field using various approaches but the accuracy is less due to the ignorance of semantic representations and the informal text in the documents.

Buddhiraja et al. [16] created an approach called POSTER where they used the two hidden layers to learn the similarity of given pair of Bug Reports.

Xie et al. [17] also used the Convolutional Neural Networks to capture the semantic representations along with domain specific information to detect the duplicate Bug Reports.

Deshmukh et al. [18] used the Convolutional Neural Networks and Long Short Term Memory for accurate and better retrieval of duplicate Bug Reports. They achieved the precision of 90 percent and the recall of 80 percent in identifying the duplicate Bug Reports. They also used Siamese Neural Networks with max margin objective for distinguishing the similar Bug Reports. Word Embedding's were used to represent the data of Bug Report. This encoded data is then passed to the Convolutional Neural Networks. They also used LSTM models because they help know the future words from the existing dependencies. Max and Mean both the pooling layers were used for reducing the outputs.

Kukkar et al. [19] also used Convolutional Neural Networks to identify the relevant features. The convolutional Layers were connected to the Similarity Measurement Layers which are then connected to the

fully connected neural layer which gives the output. Cosine Similarity was used in the Similarity Measurement layer. Fully connected layer contained tanh function along with the softmax function.

III. CHALLENGES OF DEEP LEARNING APPROACHES

A. Performance Issues

Deep Learning Based Models require lot of resources and are time-consuming as they involve the use of GPU memory and training the large dataset requires time. Deep Learning Based models require the model to be run for multiple times and each run requires lot of cost. It has been observed by one of the researcher Chen et al. [3] of how the Deep Learning Based systems were impacted in terms of time when they upgraded their models from TensorFlow 1.x to TensorFlow 2.x . They analysed the impact of deep learning projects over the performance of application and classified the issues as slow execution time, slow initialization time, out of memory, memory leak, program hang, and abnormal GPU use. Many researchers are working on reducing these issues by using the distributed model training Rhu et al. [20], using the dynamic GPU memory manager Wang et al. [21], optimizing the distribution configuration Yan et al. [22].

B. Training Dataset

Even though the Deep Learning Models are very powerful when it comes from the performance. But the models require the use of very big training dataset.

C. Complex Architecture and Scalability:

They require long training times, are very complex to understand and implement. Sometimes they also do not scale up very well [10].

IV. CONCLUSIONS

The paper classifies the main mining tasks of the Bug Report Analysis, analyzes the deep learning models being used for them. The paper also describes the deep learning approaches being used in short by various researchers. The paper apart from just mentioning the works, also analyzes the challenges of these models for the Bug Reports. The paper serves as the beginning point for the novel researchers in this field.\

From the analysis we observe that the deep learning models help captures the semantic and syntactic information of the text. Convolutional Neural Networks and the Long Short Term Memory models are among the most used models for mining the Bug Reports. Recurrent Neural Networks and Connected Neural Networks even though have been applied for various tasks but their limitations make other models more popular. We also observe that lot of opportunities exist for further improvement like the inclusion of gated units, incorporating the attention mechanisms and the use of large datasets for the training purpose.

REFERENCES

[1]. Chauhan A., Kumar R. Bug Severity Classification Using Semantic Feature with Convolution Neural Network. In: Iyer B., Deshpande P., Sharma S., Shiurkar U. (eds) Computing in Engineering and Technology. Advances in Intelligent Systems and Computing, vol 1025. Springer, Singapore. https://doi.org/10.1007/978-981-32-9515-5_31. 2020

[2]. Xiaochen Li, He Jiang, Dong Liu, Zhilei Ren, and Ge Li. 2018. *Unsupervised deep bug report summarization.* In Proceedings of the 26th Conference on Program Comprehension (ICPC '18). Association for Computing Machinery, New York, NY, USA, 144–155. DOI:https://doi.org/10.1145/3196321.3196326

[3]. Thesis on "Security Bug Report Classifification using Feature Selection, Clustering, and Deep Learning" by Tanner D. Gantzer. West Virginia University. 2019

[4]. Kumari M., Singh V.B. *An Improved Classifier Based on Entropy and Deep Learning for Bug Priority Prediction.* In: Abraham A., Cherukuri A., Melin P., Gandhi N. (eds) Intelligent Systems Design and Applications. ISDA 2018 2018. Advances in Intelligent Systems and Computing, vol 940. Springer, Cham. https://doi.org/10.1007/978-3-030-16657-1_53. 2020

[5]. Sukriti Verma and Vagisha Nidhi. *Extractive Summarization using Deep Learning* arXiv:1708.04439v2

[6]. N. Kalchbrenner, E. Grefenstette, and P. Blunsom, ``*A convolutional neural network for modelling sentences,*'' in Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics, Baltimore, MD, USA, vol. 1, Jun. 2014, pp. 655-665.

[7]. Haoran Liu, Yue Yu, Shanshan Li, Yong Guo, Deze Wang, and Xiaoguang Mao. 2020. *BugSum: Deep Context Understanding for Bug Report Summarization.* In Proceedings of the 28th International Conference on Program Comprehension ICPC '20). Association for Computing Machinery, New York, NY, USA, 94–105. DOI:https://doi.org/10.1145/3387904.3389272. 2020

[8]. Cao, Junming & Chen, Bihuan & Sun, Chao & Hu, Longjie & Peng, Xin. (2021). *Characterizing Performance Bugs in Deep Learning Systems*. 2021

[9]. H. Liang, L. Sun, M. Wang and Y. Yang, "Deep Learning With Customized Abstract Syntax Tree for Bug Localization," in *IEEE Access*, vol. 7, pp. 116309-116320, 2019, doi: 10.1109/ACCESS.2019.2936948.

[10]. Senthil Mani, Anush Sankaran, and Rahul Aralikatte. 2019. DeepTriage: *Exploring the Effectiveness of Deep Learning for Bug Triaging*. In Proceedings of the ACM India Joint International Conference on Data Science and Management of Data CoDS-COMAD '19). Association for Computing Machinery, New York, NY, USA, 171–179. DOI:https://doi.org/10.1145/3297001.3297023

[11]. Florea AC., Anvik J., Andonie R. *Parallel Implementation of a Bug Report Assignment Recommender Using Deep Learning*. In: Lintas A., Rovetta S., Verschure P., Villa A. (eds) Artificial Neural Networks and Machine Learning – ICANN 2017. ICANN 2017. Lecture Notes in Computer Science, vol 10614. Springer, Cham. https://doi.org/10.1007/978-3-319-68612-7_8. 2017

[12]. J. He, L. Xu, Y. Fan, Z. Xu, M. Yan and Y. Lei, "*Deep Learning Based Valid Bug Reports Determination and Explanation*," 2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE), 2020, pp. 184-194, doi: 10.1109/ISSRE5003.2020.00026.

[13]. An Ngoc Lam, Anh Tuan Nguyen, Hoan Anh Nguyen, and Tien N. Nguyen. 2017. *Bug localization with combination of deep learning and information retrieval.* In Proceedings of the 25th International Conference on Program Comprehension (ICPC '17). IEEE Press, 218–229. DOI: https://doi.org/10.1109/ICPC.2017.24

[14]. A. N. Lam, A. T. Nguyen, H. A. Nguyen, and T. N. Nguyen, ``Combining deep learning with information retrieval to localize buggy _les for bug reports (N)," in *Proc. 30th IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE)*, Nov. 2015, pp. 476_481

[15]. Y. Xiao, J. Keung, Q. Mi, and K. E. Bennin, ``*Bug localization with semantic and structural features using convolutional neural network and cascade forest*," in *Proc. 22nd Int. Conf. Eval. Assessment Softw. Eng.*, Christchurch, New Zealand, Jun. 2018, pp. 101_111

[16]. Sharma, Tamanna & Sangwan, Om. *Study of Information Retrieval and Machine Learning-Based Software Bug Localization Models*. 2020. 10.1007/978-981-15-0222-4_47.

[17]. A. Budhiraja, K. Dutta, R. Reddy and M. Shrivastava, "Poster: DWEN: Deep Word Embedding Network for Duplicate Bug Report Detection in Software Repositories," 2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion), 2018, pp. 193-194.

[18]. Q. Xie, Z. Wen, J. Zhu, C. Gao and Z. Zheng, "Detecting Duplicate Bug Reports with Convolutional Neural Networks," 2018 25th Asia-Pacific Software Engineering Conference (APSEC), 2018, pp. 416-425, doi: 10.1109/APSEC.2018.00056.

[19]. J. Deshmukh, K. M. Annervaz, S. Podder, S. Sengupta and N. Dubash, "Towards Accurate Duplicate Bug Retrieval Using Deep Learning Techniques," 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME), 2017, pp. 115-124, doi: 10.1109/ICSME.2017.69.

[20]. A. Kukkar, R. Mohana, Y. Kumar, A. Nayyar, M. Bilal and K. -S. Kwak, "Duplicate Bug Report Detection and Classification System Based on Deep Learning Technique," in IEEE Access, vol. 8, pp. 200749-200763, 2020, doi: 10.1109/ACCESS.2020.3033045.

[21]. Minsoo Rhu, Natalia Gimelshein, Jason Clemons, Arslan Zulfiqar, and StephenW Keckler. 2016. vDNN: Virtualized deep neural networks for scalable, memory efficient neural network design. In Proceedings of the 49th Annual IEEE/ACM International Symposium on Microarchitecture. 1–13.

[22]. Linnan Wang, Jinmian Ye, Yiyang Zhao, Wei Wu, Ang Li, Shuaiwen Leon Song, Zenglin Xu, and Tim Kraska. 2018. Superneurons: Dynamic GPU memory management for training deep neural networks. In Proceedings of the 23rd ACM SIGPLAN symposium on principles and practice of parallel programming. 41–53.

[23]. Feng Yan, Olatunji Ruwase, Yuxiong He, and Trishul Chilimbi. 2015. Performance modeling and scalability optimization of distributed deep learning systems. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 1355–1364.

