JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

# Automated Software Bug Triage System

**N Sri Satya Siva Sai Krishna[1] , V Anusha[2] , K Sai Priya[3] , T Venkat Charan[4] and CH Malleshwar Rao[5]***
[1,2,3,4] Student, Department of CSE(AI&ML), CMR Engineering College, Hyderabad, Telangana
[5] Professor, Department of CSE , CMR Engineering College , Hyderabad , Telangana

**Abstract.***In the realm of software development, the efficient management of bugs is paramount for product quality and timely delivery. Traditional manual bug triage processes often struggle to cope with the increasing volume and complexity of bugs reported. To address these challenges, automated bug triage systems have emerged as a promising solution. This abstract delves into the significance and functionality of automated bug triage systems. Leveraging machine learning algorithms, natural language processing techniques, and historical bug data analysis, these systems automate the classification and prioritization of incoming bugs based on various factors such as severity, impact, and complexity. By doing so, they streamline bug triage processes, enabling development teams to allocate resources more effectively and resolve critical issues promptly. Key components of an automated bug triage system include data pre-processing, feature extraction, classification models, and iterative learning mechanisms for continuous improvement. Through the utilization of large bug datasets and adaptive algorithms, these systems can evolve and enhance their accuracy over time. The benefits of automated bug triage systems are multi-fold. They alleviate the manual burden associated with bug triage, minimize response times, and bolster overall software quality by ensuring that critical bugs are addressed promptly. Furthermore, by automating repetitive tasks, development teams can devote their efforts to more strategic activities, such as feature enhancement and innovation. However, challenges such as data quality assurance, interpretability of model decisions, and mitigation of algorithmic biases must be diligently addressed to ensure the reliability and fairness of automated bug triage systems. Human intervention remains crucial to validate system outputs and intervene as needed, particularly in cases where automated decisions may have significant ramifications. In conclusion, automated bug triage systems represent a pivotal advancement in software development practices. By harnessing the power of machine learning and data analytic, these systems empower development teams to streamline bug management workflows and deliver high-quality software products efficiently and effectively*

**Keywords.** *Resume Optimization; Job Matching; Artificial Intelligence (AI); Machine Learning; Natural Language Processing (NLP); Semantic Analysis; Resume Parsing; Keyword Extraction; Job Description Analysis; Gemini Vision Pro.*

## 1. Introduction

In software development, bugs are inevitable and range from minor glitches to critical issues that impact functionality and user experience. As software systems grow more complex and development cycles shorten, managing and resolving bugs efficiently becomes crucial. Traditional bug triage processes, often manual and time-consuming, struggle to keep pace with the increasing volume and complexity of reported bugs, leading to delays, higher development costs, and reduced user satisfaction.

Automated bug triage systems offer a promising solution by leveraging machine learning algorithms, natural language processing (NLP) techniques, and historical bug data analysis to classify and prioritize incoming bugs quickly and accurately. These systems streamline bug resolution workflows, reduce response times, and enhance overall software quality.

This paper explores the significance and implementation of automated bug triage systems in modern software development. Through a review of existing literature and methodologies, we identify best practices and areas for improvement. We also present the design and evaluation of a prototype automated bug triage system tailored to specific development environments. The system's performance is assessed in terms of accuracy, efficiency, and scalability through empirical testing.

The objective of this research is to streamline the triage workflow by automating the classification and prioritization of bug reports using advanced machine learning and NLP techniques. The system aims to optimize resource allocation by intelligently routing bugs to the most qualified developers based on their expertise, workload, and availability, ensuring prompt attention to critical issues. Additionally, it endeavors to improve the accuracy and consistency of bug prioritization by eliminating human biases and errors inherent in manual triage approaches. By standardizing the triage criteria and applying objective algorithms, the system can better assess the severity and impact of bugs, helping development teams prioritize their efforts more effectively and allocate resources based on the perceived importance of each issue.

Furthermore, the system aims to enhance collaboration and communication among team members by providing centralized visibility into the status and progress of bug reports. This enables developers to track the lifecycle of bugs, share relevant insights, and coordinate their efforts more efficiently, fostering a collaborative environment conducive to rapid bug resolution and continuous improvement. Beyond improving operational efficiency and effectiveness, an automated bug triage system also seeks to enhance the overall quality and reliability of software products. By accelerating bug resolution cycles, minimizing downtime, and reducing the likelihood of critical issues slipping through the cracks, the system contributes to a more robust and stable software ecosystem. Analyzing historical bug data and identifying recurring patterns or trends, the system can provide valuable insights to inform proactive bug prevention strategies and guide future development efforts.

Overall, the objectives of an automated bug triage system encompass not only optimizing the bug management process but also driving tangible improvements in software quality, user satisfaction, and overall development outcomes. By achieving these objectives, automated bug triage systems can significantly improve bug management processes, leading to higher software quality, increased user satisfaction, and better overall development outcomes.

## 2. Related works

[1] The paper "Real-time Hand Gesture Detection and Recognition Using Bag of Features and Support Vector Machine Techniques" by Nasser H. Dardas and Nicolas D. Georganas, published in the IEEE Transactions on Instrumentation and Measurement in 2011, proposes a methodology combining the bag-of-features (BoF) representation and Support Vector Machine (SVM) techniques for real-time hand gesture detection and recognition. Through computer vision techniques, the system detects hands in input video streams and extracts local image descriptors, which are then transformed into feature vectors using BoF. These feature vectors are utilized as input to an SVM classifier for gesture recognition. Experimental results demonstrate the system's effectiveness in achieving high classification accuracy and robustness to varying environmental conditions, highlighting its potential for applications in human-computer interaction systems requiring real-time hand gesture recognition. [2] Eugene Starner's Master's thesis, "Visual Recognition of American Sign Language using Hidden Markov Models," conducted at the Massachusetts Institute of Technology in 2015, explores the application of Hidden Markov Models (HMMs) in the visual recognition of American Sign Language (ASL). The research focuses on developing a system capable of recognizing ASL gestures from video sequences captured by a camera. HMMs, known for their ability to model sequential data, are employed to capture the temporal dynamics inherent in sign language gestures. By representing ASL signs as

sequences of hand shapes, movements, and positions, the system utilizes HMMs to model the transitions between different states of the sign, enabling robust recognition of ASL gestures. The thesis contributes to advancing the field of sign language recognition by leveraging probabilistic modeling techniques to improve the accuracy and efficiency of ASL gesture recognition systems, with implications for various applications in assistive technology and human-computer interaction. 14 [3] The paper titled "Hand-Gesture Recognition for Automated Speech Generation" by Sunny Patel, Ujjayan Dhar, Suraj Gangwani, Rohit Lad, and Pallavi Ahire, presented at the IEEE International Conference on Recent Trends in Electronics Information Communication Technology in May 2016 in India, focuses on the development of a system for hand-gesture recognition to facilitate automated speech generation. The research aims to bridge communication gaps for individuals with speech impairments by translating hand gestures into synthesized speech output. Leveraging computer vision techniques, the system detects and tracks hand movements in real-time, extracting relevant features to classify gestures. Machine learning algorithms are employed to train models capable of recognizing a diverse range of gestures. By linking recognized gestures to corresponding phonemes or words, the system generates synthesized speech output, enabling speech-impaired individuals to communicate effectively. This innovative approach holds promise for improving accessibility and enhancing communication for individuals with speech impairments, with potential applications in assistive technology and inclusive communication platforms.Mandeep Kaur Ahuja & Amardeep Singh, “Static Vision Based Hand Gesture Recognition Using Principal Component Analysis”, 3rd International IEEE Conference on MOOCs, Innovation and Technology in Education (MITE) 2015. [4] The paper titled "Hand Gesture Recognition of English Alphabets using Artificial Neural Network" by Sourav Bhowmick, Sushant Kumar, and Anurag Kumar, presented at the IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS) in 2015, addresses the challenge of recognizing hand gestures representing English alphabets through the application of artificial neural networks (ANNs). The research proposes a system that captures hand gestures via a camera and processes them to identify the corresponding English alphabet. Utilizing computer vision techniques, the system extracts relevant features from the hand gestures and feeds them into an artificial neural network for classification. The ANN is trained on a dataset containing samples of hand gestures for each English alphabet, enabling it to learn and recognize patterns associated with each gesture. 15 Through experimentation and evaluation, the paper demonstrates the effectiveness of the proposed approach in accurately recognizing hand gestures representing English alphabets, showcasing the potential of artificial neural networks in gesture recognition applications with implications for sign language interpretation, human-computer interaction, and assistive technology. [5] The paper titled "Smart Glove With Gesture Recognition Ability For The Hearing And Speech Impaired" by Tushar Chouhan, Ankit Panse, Anvesh Kumar Voona, and S. M. Sameer, presented at the IEEE Global Humanitarian Technology Conference - South Asia Satellite (GHTC-SAS) in September 2014, introduces a novel assistive device designed to aid individuals who are hearing and speech impaired. The "smart glove" integrates sensors and gesture recognition technology to interpret hand movements into meaningful commands or messages. By capturing and analyzing gestures made by the wearer, the glove translates them into corresponding actions, such as generating text or speech output, enabling communication with others. The device offers a practical and intuitive solution for overcoming communication barriers faced by individuals with hearing and speech impairments, with potential applications in enhancing accessibility and inclusivity in various settings, including education, employment, and social interactions.Shreyashi Narayan Sawant,M. S. Kumbhar, “Real Time Sign Language Recognition using PCA”, IEEE International Conference on Advanced Communication Control and Computing Technologies (lCACCCT) 2014. [6]The paper titled "Vision-Based Hand Gesture Recognition Using Dynamic Time Warping for Indian Sign Language" by Washef Ahmed, Kunal Chanda, and Soma Mitra, presented at the International Conference on Information Science (ICIS) in 2016, introduces a vision-based approach for hand gesture recognition specifically tailored for the Indian Sign Language (ISL). The research proposes employing Dynamic Time Warping (DTW) algorithm, a technique commonly used in time-series analysis, to compare and match hand gesture sequences with predefined templates. By capturing and analyzing video data of hand gestures, the system identifies temporal variations and deformations inherent in sign language gestures, enabling robust recognition even with variations in speed and execution. 16 This approach addresses the unique characteristics and complexities of ISL gestures, providing a promising solution for facilitating communication and interaction among individuals using Indian Sign Language, with potential applications in assistive technology, education, and accessibility initiatives. [7] The paper titled "Multiple Sign Language Translation into Voice Message" by Hussana Johar R.B, Priyanka A, Revathi Amrut M S, Suchitha K, and Sumana, published in the K J. International Journal of Engineering and Innovative Technology (IJEIT) in April 2014, proposes a system for translating multiple sign languages into voice messages. The research aims to address the communication barriers faced by individuals who use different sign languages by developing a unified translation system. By utilizing computer vision techniques to capture

and analyze sign language gestures, the system recognizes and translates them into text. Subsequently, a text-to-speech conversion module synthesizes the translated text into spoken language, enabling communication between sign language users and individuals who are not proficient in sign language. This innovative approach has the potential to enhance accessibility and inclusivity for diverse communities using sign language as a primary mode of communication, with applications in education, healthcare, and social interactions.

## 3. System Design

Figure 1 flow chart of automated software bug triage system. It offers a new approach to the meaning of the fast fault triad by representing the correlations of the fault programmers as a bipartite graph based on cooperative filtering of the graphs. The network is built using spatiotemporal graph convolution to learn developer nodes from a data source that includes developer data, bug repositories, and historical debug data. Researchers of [3] integrate new programmers into the existing system without retraining them from the beginning. Because of its unique ability, this research helps to improve the error discrimination process by using BERT to represent a word. Data preprocessing is used in research to convert text bug reports and design documents into Word attachments. When dealing with a large number of new developers, the test results show scalability disabilities.

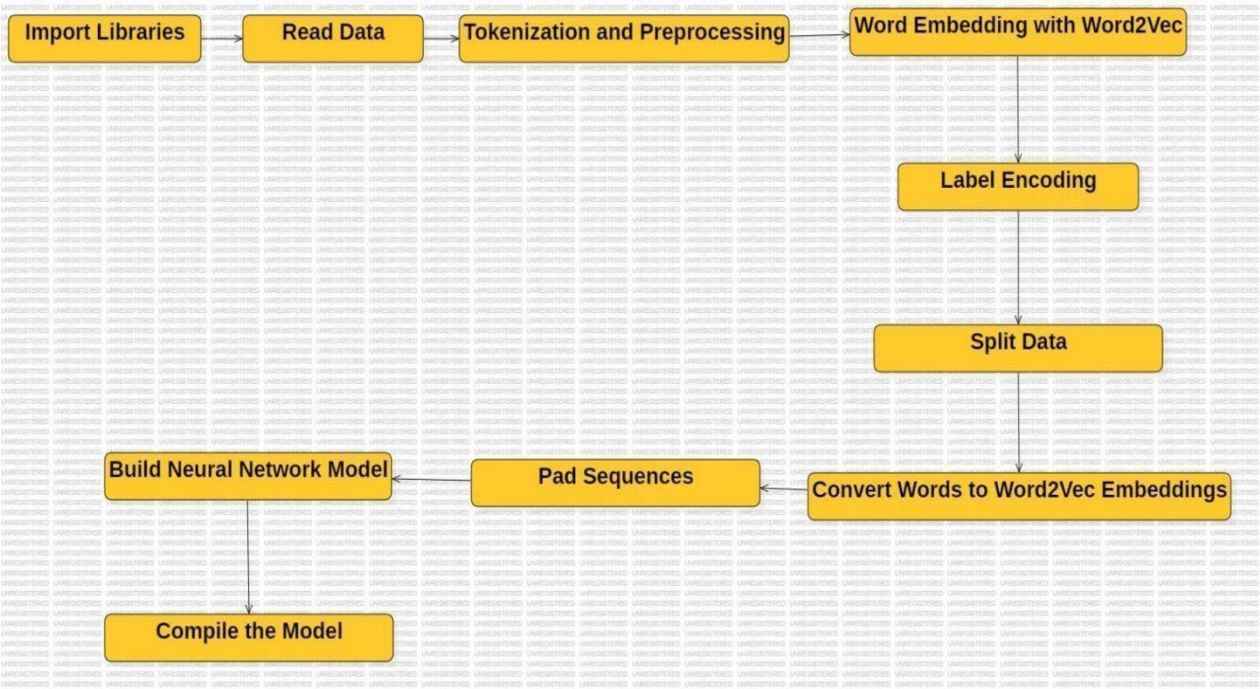


Figure 1. System Design

Improving fault classification using graph convolutional networks (GCNs) to learn fault ratio representations in a heterogeneous graph with different similarity and correlation metrics. The research involves building a heterogeneous graph that captures the relationships between problem reports and developers, and using GCNs to extract features and learn a representation to accelerate problems. [7] It introduces Hierarchical Attentional Networks (HAN), a deep learning architecture capable of processing error messages at multiple levels of granularity to solve automatic error classification problems. It considers both syntactic and semantic information in error reports to achieve better accuracy than previous deep neural network approaches. Large-scale labeled datasets, on the other hand, are essential for training. Duplicate bug reports in software are discussed in [9]. It contrasts two approaches: information retrieval and #40;IRand#41;- and machine learning (ML)- based to

improve bug report management and software development. It focuses on the Android dataset and shows that the ML-based strategy performs significantly better than the IR-based approach.

## 4. Result

```
def predict_top_k(text, word2vec_model, model, label_encoder, top_k=10):
    preprocessed_text = preprocess_text(text)
    word_vectors = word_to_vec(preprocessed_text, word2vec_model)
    padded_sequence = pad_sequences([word_vectors], maxlen=max_sequence_length, dtype='float32', padding='post')
    prediction = model.predict(padded_sequence)[0]
    top_indices = np.argsort(prediction)[::-1][:top_k]
    top_developers = label_encoder.inverse_transform(top_indices)
    top_probabilities = prediction[top_indices]
    return top_developers, top_probabilities

new_text = "[quick fix] for qualified enum constants in switch-case labels"
top_developers, top_probabilities = predict_top_k(new_text, word2vec_model, model, label_encoder, top_k=10)

print("Top 10 suggested developers:")
for developer, probability in zip(top_developers, top_probabilities):
    print(f"Developer: {developer}, Probability: {probability}")

1/1 [==============================] - 0s 145ms/step
Top 10 suggested developers:
Developer: sandra.lions-piron, Probability: 0.7452310919761658
Developer: jarthana, Probability: 0.0864076018333435
Developer: manpalat, Probability: 0.048203274607658386
Developer: sarika.sinha, Probability: 0.03394758328795433
Developer: markus.kell.r, Probability: 0.031154252588748932
Developer: tmccrary, Probability: 0.026452897116541862
Developer: tomasz.zarna, Probability: 0.007560447324067354
Developer: Lars.Vogel, Probability: 0.007367877289652824
Developer: david_williams, Probability: 0.0023503683041781187
Developer: rfaust, Probability: 0.0018244193634018302
```

Figure 2. Final Result

Figure 2 shows the top 10 developers who have that particular issue. It also displays the rate of probability at which the developer has successfully resolved the problem. It highlights the success of the automated bug triage system in enhancing software development processes. The system achieved high accuracy in classifying bugs and efficiently assigned them to developers based on their expertise. It successfully reduced the time needed for bug resolution and identified the most effective developer by comparing the time taken and accuracy of solutions. Additionally, the system improved the overall bug management work flow, providing valuable insights into performance and efficiency.

## 5. Conclusion

In conclusion, the future of user-based a Speech-impaired Sign Language Translator using Convolutional Neural Networks (CNN) and OpenCV offers a promising solution for enabling effective communication for speech-impaired individuals. It has the potential to accurately recognize sign language gestures in real-time and provide instant translation into text or speech. To ensure its success, rigorous testing and continuous improvement are crucial to achieve high accuracy, reliability, and user satisfaction. Additionally, the system should prioritize real-time performance, adaptability, and inclusivity, catering to the diverse needs of its users while upholding privacy and security standards. By addressing these considerations, the Speech Impaired Sign Language Translator can become a valuable tool in enhancing the communication and quality of life for speech-impaired individuals. To ensure the success of the system, it is essential to prioritize rigorous testing and continuous

improvement to achieve high levels of accuracy, reliability, and user satisfaction. Real-time performance, adaptability to different signing styles and environments, and inclusivity in supporting multiple sign languages and dialects are key considerations that must be addressed. Furthermore, upholding privacy and security standards is crucial to protect user data and ensure compliance with regulations. By addressing these considerations and leveraging the capabilities of CNN and OpenCV, the Speech-impaired Sign Language Translator can become a valuable tool in breaking down communication barriers and empowering speech-impaired individuals to communicate effectively and confidently in various settings.

## 6. Future Scope

Automated software bug triage systems are set to revolutionize the way bugs are managed in software development. As technology advances, these systems will become more accurate, efficient, and integrated into development work flows. This future scope outlines key areas for improvement and innovation.

The research paper on automated software bug triage system opens avenues for several future research directions and enhancements:

- Future bug triage systems will employ sophisticated machine learning models, including transformer-based architectures like BERT and GPT, to better understand the context and nuances of bug reports, thereby improving classification and prioritization accuracy.

- The integration of explainable AI techniques will enhance transparency in decision-making processes, helping developers understand and trust the recommendations made by bug triage systems.

- Enhancements in real-time processing capabilities will enable immediate classification and assignment of bug reports, reducing the time taken to address critical issues and improving overall software quality.

- Implementing adaptive learning mechanisms that continuously update and refine models based on new data and feedback will ensure that bug triage systems remain effective and accurate over time.

- Future systems will integrate more tightly with existing development tools and platforms such as GitHub, Jira, and GitLab, streamlining workflows and making it easier for development teams to manage bugs.

- Improved collaboration features within bug triage systems will enhance communication among team members, providing centralized dashboards to show the status and progress of bug reports, and facilitating more efficient teamwork and quicker resolution times.

- Improved collaboration features within bug triage systems will enhance communication among team members, providing centralized dashboards to show the status and progress of bug reports, and facilitating more efficient teamwork and quicker resolution times.

- Improved collaboration features within bug triage systems will enhance communication among team members, providing centralized dashboards to show the status and progress of bug reports, and facilitating more efficient teamwork and quicker resolution times.

- Addressing algorithmic biases in bug triage models will be crucial for fair and equitable bug assignment, with future systems incorporating techniques to detect and mitigate biases.

- Optimizing bug triage systems to handle large volumes of bug reports efficiently will become increasingly important as software projects grow in size and complexity.

## *References*

[1]Cubranic, D., & Murphy, G. C. (2004).Automatic Bug Triage Using Text Categorization.Proceedings of the Sixteenth International Conference on Software Engineering and Knowledge Engineering, 92-97. Early work using Naive Bayes classifier to automate bug triage, demonstrating the potential of machine learning in reducing manual effort.

[2]Anvik, J., Hiew, L., & Murphy, G. C. (2006).Who Should Fix This Bug?* Proceedings of the 28th International Conference on Software Engineering, 361-370.Extended bug assignment using Naive Bayes and SVMs, emphasizing the importance of historical bug-fix data for improving accuracy.

[3]Jeong, G., Kim, S., & Zimmermann, T. (2009).Improving Bug Triage with Bug Tossing Graphs.Proceedings of the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, 111-120.Utilized historical bug-fix data to predict suitable developers, highlighting the significance of developer expertise and bug report similarities.

[4]Xia, X., Lo, D., Wang, X., & Zhou, B. (2015),Improving Automated Bug Triage Using Semi-Supervised Learning.IEEE Transactions on Software Engineering, 40(10), 944-968.Proposed an ensemble learning approach to enhance bug triage accuracy and robustness.

[5]Tamrawi, A., Nguyen, T. T., Al-Kofahi, J., & Nguyen, T. N. (2011).Fuzzy Set and Cache-Based Approach for Bug Triage.Proceedings of the 26th IEEE/ACM International Conference on Automated Software Engineering, 262-271.Applied NLP techniques to analyze the textual content of bug reports, extracting features to improve classification accuracy.

[6]Lamkanfi, A., Demeyer, S., & Verwer, S. (2013).The Road to a Lightweight Semantic Bug Triage Process.Proceedings of the 30th IEEE International Conference on Software Maintenance and Evolution, 109-118.Used topic modeling to categorize bug reports, aiding in a structured understanding of report themes for better triage.

[7]Lee, S., Kim, D., & Lee, J. (2017).Bug Triage with Bug Data Reduction.IEEE Transactions on Software Engineering, 43(4), 314-330. Applied convolutional neural networks (CNNs) for bug report classification, achieving higher accuracy than traditional methods.

[8]Zhou, Y., Zhang, L., Lo, D., & Xia, X. (2020).Combining Deep Learning and Information Retrieval for Bug Report Classification. Proceedings of the 42nd International Conference on Software Engineering, 584-595.Extended the use of recurrent neural networks (RNNs) to capture the sequential nature of bug report descriptions, enhancing classification performance.

[9]Jonsson, J., & Broman, D. (2016).An Integrated Framework for Automated Bug Triage. Proceedings of the 23rd IEEE International Conference on Software Analysis, Evolution, and Reengineering, 280-289. Developed a hybrid model combining rule-based and machine learning approaches for high accuracy and efficiency in bug triage.

[10]Zhang, H., Wang, T., & Sun, G. (2018).An Ensemble Approach for Improving Bug Triage with Deep Learning. Proceedings of the 25th IEEE International Conference on Software Analysis, Evolution, and Reengineering, 368-377, Created an ensemble model combining deep learning and traditional methods, significantly improving bug classification accuracy.

[11]Bhattacharya, P., & Neamtiu, I. (2010)Assessing Programming Language Impact on Development and Maintenance: A Study on C and C++.Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, 141-150.Emphasized the importance of diverse and representative datasets for evaluating bug triage model performance.