International Journal of Advanced Multidisciplinary Research
ISSN: 2393-8870
www.ijarm.com
(A Peer Reviewed, Referred, Indexed and Open Access Journal)
DOI: 10.22192/ijamr Volume 11, Issue 3 -2024

Research Article

DOI: http://dx.doi.org/10.22192/ijamr.2024.11.03.005

Exploring the Use of ChatGPT for Resolving Programming Bugs

Hussaini Dan'azumi[1], Dr. Yakubu Bala Mohammed[2*], and Mahmood Saidu Badara[3]

[1]Department of Computer Science, Abubakar Tatari Ali Polytechnic, Bauchi, Nigeria, Bauchi-State, Jos Road 0094, Nigeria.
E-mail: *hussainidanazumi@gmail.com*

[2*]Department of Computer Science, Abubakar Tatari Ali Polytechnic, Bauchi, Nigeria, Bauchi-State, Jos Road 0094, Nigeria.
E-mail: *mohammedbala0079@gmail.com*

[3]Department of Computer Science, Abubakar Tatari Ali Polytechnic, Bauchi, Nigeria, Bauchi-State, Jos Road 0094, Nigeria.
E-mail: *Mahmood_saidu@yahoo.com*

Abstract

ChatGPT is an advanced language model that has been gaining attention in the natural language processing field. However, its functionalities go beyond language-related tasks. ChatGPT can also serve as a robust tool for debugging software code. Debugging holds crucial significance in the software development process as bugs, or code errors, can significantly impact the functionality and security of applications. The process of identifying and rectifying bugs using traditional debugging approaches can be labour-intensive and time-consuming, typically requiring the expertise of skilled developers. With the growing complexity of software applications, there is an increasing demand for efficient and precise debugging solutions. Thus, the purpose of this study is to investigate the effectiveness of ChatGPT in identifying, predicting, explaining, and resolving programming bugs. Findings of the study revealed that ChatGPT can to analyze and comprehend code. Also, the review results discovered that ChatGPT has the potential to streamline the debugging process, making it more suitable for developers with varying experience levels. Lastly, the study offers insights into integrating ChatGPT into the software development workflow, and proposed directions for upcoming studies.

Keywords

ChatGPT,
Programming Bugs,
Debugging techniques,
Bugs Prediction.

1. Introduction

Nowadays, computer applications (programs) play a significant role in addressing real-world problems (Modi et al., 2013). However, Bugs in computer code can lead to serious problems, such asminor software crashes,data loss, and security vulnerability. Surameery and Shakor (2023) and (Zhong & Su, 2015) in their studies argued that debugging code i.e., the "process of searching and fixing program bugs is the most important aspect of software development that can be complex, and time-consuming". Debugging is the process of spotting and correcting errors, or bugs, in software code (Morovati et al., 2023). Bugs can happen because of different stuff, like typos, wrong syntax, wrong logic, or unexpected inputs.Bharadwaj and Parker (2023) in their work stressed that bugs can make the program act weird, give the wrong answers, or even crash. The authors argued that "it is a critical part of the software development process, as it ensures that software applications are functioning correctly and efficiently" (Nathalia et al., 2023). Though, studies (Dakhel et al., 2023; Nistor et al., 2013; Xuan et al., 2016) have shown that software engineers have done a lot of research in an attempt to limit the number of bugs found in the software development process in order to make codes better, however, bugs (i.e., errors) still remain issue for software developers (DeLiema et al., 2023; Kang et al., 2023; Xin & Reiss, 2017).

Recent advancements in artificial intelligence (AI) have offers new opportunities for automating different software development tasks e.g., finding and fixing programming bugs i.e., debugging (Nagwani & Suri, 2023). Jesse et al. (2023) in their work contended that nowadays, "developers can use AI coding tools that get their strength from models trained on huge amounts of open-source code to trace and fix programming bugs" such as Copilot and ChatGPT. Copilot is an AI-powered coding assistant developed by GitHub and OpenAI. It uses machine learning models trained on vast amounts of open-source code to assist developers in writing code more efficiently and effectively. It has the ability to suggest code completions, helps with writing functions, and provides context-aware code suggestions as developers write code in their Integrated Development Environment (Perry et al., 2023). While ChatGPT on the other hand ChatGPT is a cutting-edge language model built on the GPT framework (Zhang et al., 2023). Since its launchon November 30, 2022, ChatGPT has gained immense popularity as users explore its features and get more accustomed to using it (Yilmaz & Yilmaz, 2023). It possesses numerous undiscovered capabilities that could bring about major changes in various sectors, from online shopping to mental healthcare. Though these capabilities are still developing, they have the potential to revolutionize our lifestyles, professions, and interactions. Additionally, ChatGPT is capable of handling a diverse array of tasks, including debugging. Thus, the purpose of this research is to examine the potential benefits of employing AI-based debugging tools, specifically ChatGPT to enhance the efficiency and accuracy of the debugging process.

The review illustrates the viability and efficacy of leveraging ChatGPT for identifying and rectifying errors in computer code. The review results will offer insights into the capabilities of AI within software development and will inform the creation of future AI-driven tools for debugging code.

2. Theoretical Background

2.1 ChatGPT as Debugging Tool

In recent past, there has been a lot of focus on Large Language Models (LLMs) due to their robustness in processing programming languages for different Software Engineering (SE) tasks (Zhang et al., 2023). These LLMs usually go through a pre-training-and-finetuning process [12, 49], meaning they're first trained with self-supervised training on a big unlabelled collection to get general knowledge, and then adjusted with supervised training on a smaller labelled collection to fit a particular task(Liu et al., 2024).

Among the various LLMs, ChatGPT is considered by many as one of the most widely used language models, and it's being looked at by researchers in various fields, such as code summarization, code generation, and test generation(Sun et al., 2023).

Tufano et al. (2019), claimed that ChatGPT is a prompt-based LLM with reinforcement learning from human feedback, allowing it to engage with users through dialogues that mimic human interactions. Recently, ChatGPT has been getting a lot of attention for its remarkable ability to understand and respond to human-initiated conversations. In the context of Automated Program Repair (APR), ChatGPT has demonstrated exceptional performance in resolving issues in popular datasets such as “Defects4J” Plein et al. (2023) and “QuixBugs” Haque and Li (2023).

Sobania et al. (2023) in their work analyze how ChatGPT repairs programs by both making single requests and engaging in further discussions with it. Additionally, Cao et al. (2023) examined the effectiveness of ChatGPTs in fixing deep learning programs. The authors found that ChatGPT has the ability to trace and fix bugs in various deep learning programs. Also, Xia and Zhang (2023) in their study investigated the efficacy of Chatbot, specifically ChatGPT in tracing and fixing program bugs. The authors introduce an APR method based on ChatGPT that maximizes conversations by providing immediate feedback on previous patches. Surameery and Shakor (2023) and Biswas (2023) in their studies argued that ChatGPT has the capability to identify bugs in programs, isolate the bugs, and fix them in a more robust and flexible manner compared to other traditional debugging approaches. The authors highlight some key features of ChatGPT that could be helpful in identifying and resolving bugs in computer programs. These features encompass; i) Natural Language Processing (NLP) Skills. The authors claimed that ChatGPT has the ability to boast advanced NLP abilities, such as understanding and generating text that resembles human language, which may be beneficial for code analysis as it enables the model to grasp the purpose behind the code and spot potential bugs based on the language used, ii) Comprehensive Knowledge Representation i.e., ChatGPT has undergone extensive training on a wide array of text data, including details about software development and programming languages. Thus, enabling it to have a thorough understanding of the knowledge and principles associated with software development, which can be utilized to detect and rectify bugs in code, iii) Pattern Recognition Identification i.e., ChatGPT excels in recognizing patterns within text data, which aids in bug detection in code, and pinpoint recurring patterns in code often linked to specific bug types, iv) Error Refinement i.e., With its extensive training on large volumes of text data, ChatGPT can propose corrections to code, including bug fixes, which streamlines the debugging process, reducing the time and effort needed to locate and resolve bugs, v) Adaptability i.e., ChatGPT can generalize from its training dataset to new onesand other unseen examples. Thus, making it a valuable tool for debugging code as it enables the model to identify bugs in fresh code based on its prior knowledge.

2.2 ChatGPT Versus Traditional Debugging Methods

ChatGPT is a model for natural language processing, which means it's trained to grasp human language(Biswas, 2023). However, it can also be trained on programming languages and the syntax used in software code. For instance, when faced with a piece of code, ChatGPT can analyze it and spot potential problems(Liu et al., 2024). It’s capable of detecting various errors in code, including syntax, logic, and semantics.Syntax errors involve mistakes in the way code is written, such as missing or extra punctuation, and incorrect syntax(Reiche & Leidner, 2023). While logical errors occur when code doesn't perform as expected, even if it's written correctly(Liu et al., 2023). Semantic errors happen when the code is technically accurate but doesn't produce the desired results(Yilmaz & Yilmaz, 2023).

ChatGPT can serve as a debugger in multiple ways. It can aid in understanding code by responding to queries about syntax, function

behaviour, or code structure. For instance, a developer can ask ChatGPT about a specific function's definition or usage, and it can offer a detailed explanation to help the developer use the function correctly. It can assist in identifying and rectifying syntax errors, such as missing brackets or semicolons, by suggesting corrections or indicating where the error lies(Rahman & Watanobe, 2023).Furthermore, ChatGPT can help pinpoint the underlying cause of issues by explaining error messages, providing context on the code's operating environment, or proposing potential solutions to a problem(Meyer et al., 2023).

In the usual debugging process using traditional methods, once the error is found, developers can proceed to apply a solution to address the problem. Making changes to the code, adjusting settings, and/or making other modifications to the software are all steps involved in fixing bugs (Do et al., 2018).Benton et al. (2021) and Ghosh and Singh (2020) in their works stressed that "traditional debugging often includes a significant amount of trial and error, and it can be frustrating and time-consuming". The authors argued that in a typical debugging process (i.e., traditional methods), a successful debugging and bug-fixing process "requires a combination of technical expertise, teamwork, and careful attention to details". Thus, making the process cumbersome as shown in **Figure 1**.

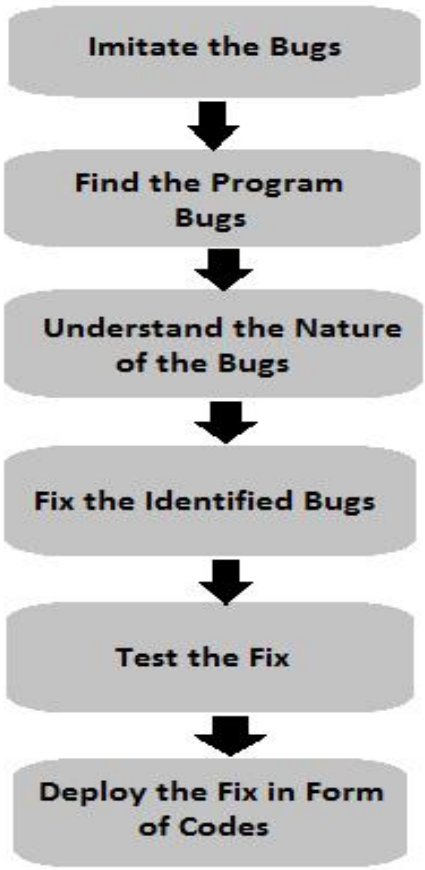

Figure 1: Traditional Debugging Methods (Haque & Li, 2023).

In contrast to the traditional debugging process, Das et al. (2024) and Haque and Li (2023) argued that AI debugging tools, specifically ChatGPT have the skills to identify the main reasons for problems (i.e., bugs in programs) by “explaining the error messages, providing context on the environment in which the code is running, and even suggest potential solutions to the identified problems". The authors highlight some general steps which ChatGPT might take to identify, analyze, and debug a piece of programming code and fix it as shown in **Figure 2.**

Figure 2: ChatGPT Debugging Approach (Haque & Li, 2023).

As shown in **Figure 2**, the debugging process via ChatGPT consists of at least seven stages as per (Haque & Li, 2023). i)The Input phase, in the input phase, the programmer provides the code to ChatGPT alongside a description of the problem they're facing. ii) Analysis phase, in the second stage, ChatGPT examines the code using natural language processing and machine learning methods. It identifies all possible issues that are likely the cause of the bugs such as syntax errors or logical discrepancies and creates a list of potential solutions, iii) Ranking phase, in the ranking stage, ChatGPT ranks the potential solutions based on how relevant and likely they are to succeed. It might take into account factors such as the programmer's past actions, code patterns, and the success rates of similar fixes, iv) Suggestion phase, where ChatGPT generates a suggestion for resolving the bug, usually in natural language. The suggestion might involve altering the code itself, or it could propose changes to the environment or configuration settings, v) In the Feedback phase, here the programmer assesses the suggestion and gives feedback to ChatGPT. The coders might approve the suggestion, tweak it, or reject it entirely. This feedback helps enhance the accuracy of future suggestions, vi) Integration phase, in this stage, if the programmer approves the suggestion given by ChatGPT in the fifth step, they incorporate the proposed fix into their codebase. Additionally, ChatGPT might also offer tools for automating the integration process, such as code restructuring or testing frameworks that can enhance codes i.e., the enhancement phase.

Literature (Surameery & Shakor, 2023; Xia & Zhang, 2023; Xin & Reiss, 2017) have shown that both ChatGPT and traditional debugging tools have their own strengths and weaknesses. The best approach to solve a specific programming bug will depend on the particular circumstances of the bug and the developer's experience and preferences. Debugging tools like integrated development environments (IDEs) and debuggers offer a variety of features to help developers identify and fix bugs, such as breakpoints, variable inspection, and trace analysis. However, these tools can be complex to use and may require significant expertise to fully utilize. On the flip side, ChatGPT offers a more accessible and intuitive method for solving programming bugs. Its natural language processing and knowledge representation capabilities enable it to analyze code snippets and provide explanations for bugs in a manner that developers can easily comprehend. This can be especially helpful for identifying and fixing more complex bugs. The strengths of ChatGPT as a debugging tool compared to other conventional debugging tools are offered in **Table 1.**

Table 1: Comparisons between ChatGPT and Conventional Debugging Tools

Ability	Explanation
Cost	Traditional debugging tools and methods, like IDEs and debuggers, might come with a high price tag, and difficult to use, whereas ChatGPT is commonly available as a cloud-based service with a pricing model that offers more flexibility.
Speediness	ChatGPT can offer speedy and effective bug explanations and predictions, while traditional debugging tools may take longer to execute and deliver results.
Precision	The precision of ChatGPT regarding bug predictions and explanations may be influenced by the quality of its training data, whereas “traditional debugging tools and techniques might provide a greater level of accuracy, but require a deeper understanding of the codes”.
Flexibility	Traditional debugging tools have the potential for extensive customization, while ChatGPT is structured to function immediately and might not provide equivalent customization options.
User-friendliness	ChatGPT has the ability to generate natural language,which makes it simple for developers to grasp its outcomes, while traditional debugging tools may be more intricate and challenging to utilize.
Incorporation with other existing tools	Conventional debugging tools can be merged with other tools and systems, whereas ChatGPT might not provide an equivalent level of integration.
Scalability	ChatGPT have the ability to debug code on a large scale, whereas traditional debugging tools might face challenges in handling extensive and intricate codebases.

3. Results and Discussions

In this section, results regarding the effectiveness of ChatGPT in identifying, isolating, and fixing programming bugs from prior studies were offered in the following subsections.

3.1 ChatGPT as Bug Explainer

Regarding bug explanation, the study results revealed that ChatGPT can assist in elucidating bugs, detailing why a specific portion of code triggers a bug and suggesting solutions for rectification. This aids in enhancing comprehension of bugs and offers insights into averting similar issues in the future. This finding is in line with the findings of Lau and Guo (2023), Busch et al. (2023), and Surameery and Shakor (2023) who argued that "ChatGPT can provide explanations for programming bugs by using its knowledge representation and natural language generation capabilities". Furthermore, in explicating programming bugs, ChatGPT leverages its knowledge representation and natural language generation capabilities. Upon detecting a bug in code, ChatGPT utilizes its comprehension of the code and the connections between the code and the bug to craft an explanation. This elucidation aids code developers in grasping the root cause of the bug and its resolution(MacNeil et al., 2024).

3.2 ChatGPT as Bug Predictor

Concerning bug prediction, the review results revealed that ChatGPT has the capability to anticipate the likelihood of bugs in fresh code, drawing on its comprehension of the connections between code and bugs. This proves valuable in spotting bugs early in the development phase, preventing them from becoming more challenging and costly to rectify. This result is reinforced by the findings of Chen et al. (2024) who stressed that regarding bug prediction, ChatGPT can harness its skill to scrutinize and comprehend code snippets. Thus, allowing the model to utilize its knowledge representation and pattern recognition capacities to detect potential bugs in new code, relying on its training data.

The review also discovered that the efficacy of ChatGPT in programming bug prediction is contingent on the quality of the training data and system design. If the training data encompasses a comprehensive array of code snippets and bugs, the model can cultivate a robust understanding of the interplay between code and bugs. This, in turn, results in precise predictions and aids in early bug identification during the development process(Hoq et al., 2023).

3.3 ChatGPT as Debugging Aid

Findings of the study discovered that ChatGPT can assist in providing suggestions and corrections to codeby understanding the connections between code and bugs. This can streamline the debugging process and lessen the time and effort needed to locate and rectify bugs. It can also aid in debugging programming codes through its natural language processing (NLP) abilities, knowledge representation, and pattern recognition skills. This result is in agreement with the findings of Surameery and Shakor (2023) who argued that once ChatGPT is trained, the model can offer suggestions and corrections to code by comprehending the relationships between code and bugs. The authors stressed that if a bug exists in the code, ChatGPT can propose a correction based on its training data. These suggestions may be drawn from its familiarity with programming languages, typical bug patterns, and software development best practices. Thus,the effectiveness of employing ChatGPT for debugging hinges on factors such as the quality of the training data, system design, and the specific programming bugs targeted. Furthermore, the review found that the utilization of ChatGPT for debugging assistance is still an emerging field of study, requiring further investigation to fully understand its capabilities and limitations.

3.3 ChatGPT as Bug Predictor

Concerning bugs prediction, the review results revealed that ChatGPT has the capability to anticipate the likelihood of bugs in fresh code, drawing on its comprehension of the connections

between code and bugs. This proves valuable in spotting bugs early in the development phase, preventing them from becoming more challenging and costly to rectify. This result is supported by the findings of Chen et al. (2024) who stressed that regarding bug prediction, ChatGPT can harness its skill to scrutinize and comprehend code snippets. Thus, allowing the model to utilize its knowledge representation and pattern recognition capacities to detect potential bugs in new code, relying on its training data.

The review also discovered that the efficacy of ChatGPT in programming bug prediction is contingent on the quality of the training data and system design. If the training data encompasses a comprehensive array of code snippets and bugs, the model can cultivate a robust understanding of the interplay between code and bugs. This, in turn, results in precise predictions and aids in early bug identification during the development process (Hoq et al., 2023).

4. Conclusion

In conclusion, ChatGPT can contribute to resolving programming bugs by offering debugging assistance, predicting bugs, and explaining their causes. Its capacity to analyze and comprehend code snippets, combined with its knowledge representation and natural language generation capabilities, renders it suitable for these tasks. However, it's important to note that while ChatGPT can be helpful in addressing programming bugs, it's not flawless. Also, the accuracy of its outcomes relies on the quality of the training data and the system's design. Additionally, it's essential to employ other debugging tools and methods to verify ChatGPT predictions and explanations, and to ensure bug-free code. Thus, ChatGPT should be viewed as one component of a comprehensive debugging toolkit, that can be used alongside other tools and techniques to achieve optimal results. By leveraging the strengths of ChatGPT alongside those of other debugging tools, developers can gain a deeper understanding of their code and effectively identify and rectify bugs. The utilization of ChatGPT for addressing programming bugs shows promise, yet further research is required to fully assess its capabilities and limitations.

References

Benton, S., Li, X., Lou, Y., & Zhang, L. (2021). Evaluating and improving unified debugging. *IEEE Transactions on Software Engineering*, *48*(11), 4692-4716.

Bharadwaj, R., & Parker, I. (2023, June). Double-edged sword of LLMs: mitigating security risks of AI-generated code. In *Disruptive Technologies in Information Sciences VII* (Vol. 12542, pp. 141-146). SPIE.

Biswas, S. (2023). Role of ChatGPT in Computer Programming.: ChatGPT in Computer Programming. *Mesopotamian Journal of Computer Science*, *2023*, 8-16.

Busch, D., Nolte, G., Bainczyk, A., & Steffen, B. (2023, October). ChatGPT in the loop: a natural language extension for domain-specific modeling languages. In *International Conference on Bridging the Gap between AI and Reality* (pp. 375-390). Cham: Springer Nature Switzerland.

Cao, J., Li, M., Wen, M., & Cheung, S.-c. (2023). A study on prompt design, advantages and limitations of chatgpt for deep learning program repair. *arXiv preprint arXiv:2304.08191*.

Chen, M., Li, Y., & Xu, Q. (2024). HiBug: On Human-Interpretable Model Debug. *Advances in Neural Information Processing Systems*, *36*.

Dakhel, A. M., Majdinasab, V., Nikanjam, A., Khomh, F., Desmarais, M. C., & Jiang, Z. M. J. (2023). Github copilot ai pair programmer: Asset or liability? *Journal of Systems and Software*, *203*, 111734.

Das, J. K., Mondal, S., & Roy, C. K. (2024). Investigating the Utility of ChatGPT in the Issue Tracking System: An Exploratory Study. *arXiv preprint arXiv:2402.03735*.

DeLiema, D., Kwon, Y. A., Chisholm, A., Williams, I., Dahn, M., Flood, V. J., Abrahamson, D., & Steen, F. F. (2023). A multi-dimensional framework for documenting students' heterogeneous experiences with programming bugs. *Cognition and Instruction*, *41*(2), 158-200.

Do, L. N. Q., Krüger, S., Hill, P., Ali, K., & Bodden, E. (2018). Debugging static analysis. *IEEE Transactions on Software Engineering*, *46*(7), 697-709.

Ghosh, D., & Singh, J. (2020). A systematic review on program debugging techniques. *Smart Computing Paradigms: New Progresses and Challenges: Proceedings of ICACNI 2018, Volume 2*, 193-199.

Haque, M. A., & Li, S. (2023). The Potential Use of ChatGPT for Debugging and Bug Fixing. *EAI Endorsed Transactions on AI and Robotics*, *2*(1), e4-e4.

Hoq, M., Shi, Y., Leinonen, J., Babalola, D., Lynch, C., & Akram, B. (2023). Detecting chatgpt-generated code in a cs1 course. In *Workshop on empowering education with llms-the next-gen interface and content generation. 2(1)*, 221-238.

Jesse, K., Ahmed, T., Devanbu, P. T., & Morgan, E. (2023). Large Language Models and Simple, Stupid Bugs. *arXiv preprint arXiv:2303.11455*.

Kang, S., Yoon, J., & Yoo, S. (2023, May). Large language models are few-shot testers: Exploring llm-based general bug reproduction. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, (pp. 2312-2323). IEEE.

Lau, S., & Guo, P. (2023, August). From" Ban it till we understand it" to" Resistance is futile": How university programming instructors plan to adapt as more students use AI code generation and explanation tools such as ChatGPT and GitHub Copilot. In *Proceedings of the 2023 ACM Conference on International Computing Education Research-Volume 1* (pp. 106-121).

Liu, J., Xia, C. S., Wang, Y., & Zhang, L. (2024). Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems*, *36*.

Liu, Y., Le-Cong, T., Widyasari, R., Tantithamthavorn, C., Li, L., Le, X.-B. D., & Lo, D. (2023). Refining ChatGPT-generated code: Characterizing and mitigating code quality issues. *ACM Transactions on Software Engineering and Methodology1(4)*, 211-226.

MacNeil, S., Denny, P., Tran, A., Leinonen, J., Bernstein, S., Hellas, A., ... & Kim, J. (2024, January). Decoding Logic Errors: A Comparative Study on Bug Detection by Students and Large Language Models. In *Proceedings of the 26th Australasian Computing Education Conference* (pp. 11-18).

Meyer, J. G., Urbanowicz, R. J., Martin, P. C., O'Connor, K., Li, R., Peng, P.-C., Bright, T. J., Tatonetti, N., Won, K. J., & Gonzalez-Hernandez, G. (2023). ChatGPT and large language models in academia: opportunities and challenges. *BioData Mining*, *16*(1), 20.

Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., & Rajarajan, M. (2013). A survey of intrusion detection techniques in cloud. *Journal of network and computer applications*, *36*(1), 42-57.

Morovati, M. M., Nikanjam, A., Khomh, F., & Jiang, Z. M. (2023). Bugs in machine learning-based systems: a faultload benchmark. *Empirical Software Engineering*, *28*(3), 62.

Nagwani, N. K., & Suri, J. S. (2023). An artificial intelligence framework on software bug triaging, technological evolution, and future challenges: A review. *International Journal of Information Management Data Insights*, *3*(1), 100153.

Nathalia, N., Paulo, A., & Donald, C. (2023, September). Artificial intelligence vs. software engineers: An empirical study on performance and efficiency using chatgpt. In *Proceedings of the 33rd Annual International Conference on Computer Science and Software Engineering* (pp. 24-33).

Nistor, A., Jiang, T., & Tan, L. (2013, May). Discovering, reporting, and fixing performance bugs. In *2013 10th working conference on mining software repositories (MSR)* (pp. 237-246). IEEE.

Perry, N., Srivastava, M., Kumar, D., & Boneh, D. (2023, November). Do users write more insecure code with AI assistants?. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security* (pp. 2785-2799).

Plein, L., Ouédraogo, W. C., Klein, J., & Bissyandé, T. F. (2023). Automatic generation of test cases based on bug reports: a feasibility study with large language models. *arXiv preprint arXiv:2310.06320*.

Rahman, M. M., & Watanobe, Y. (2023). ChatGPT for education and research: Opportunities, threats, and strategies. *Applied Sciences*, *13*(9), 5783.

Reiche, M., & Leidner, J. L. (2023, September). Bridging the programming skill gap with ChatGPT: A machine learning project with business students. In *European Conference on Artificial Intelligence* (pp. 439-446). Cham: Springer Nature Switzerland.

Sobania, D., Briesch, M., Hanna, C., & Petke, J. (2023). An analysis of the automatic bug fixing performance of chatgpt. *arXiv preprint arXiv:2301.08653*.

Sun, W., Fang, C., You, Y., Miao, Y., Liu, Y., Li, Y., Deng, G., Huang, S., Chen, Y., & Zhang, Q. (2023). Automatic Code Summarization via ChatGPT: How Far Are We? *arXiv preprint arXiv:2305.12865*.

Surameery, N. M. S., & Shakor, M. Y. (2023). Use chat gpt to solve programming bugs. *International Journal of Information Technology & Computer Engineering (IJITC) ISSN: 2455-5290*, *3*(01), 17-22.

Tufano, M., Watson, C., Bavota, G., Penta, M. D., White, M., & Poshyvanyk, D. (2019). An empirical study on learning bug-fixing patches in the wild via neural machine translation. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *28*(4), 1-29.

Xia, C. S., & Zhang, L. (2023). Keep the Conversation Going: Fixing 162 out of 337 bugs for $0.42 each using ChatGPT. *arXiv preprint arXiv:2304.00385*.

Xin, Q., & Reiss, S. P. (2017, October). Leveraging syntax-related code for automated program repair. In *2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)* (pp. 660-670). IEEE.

Xuan, J., Martinez, M., Demarco, F., Clement, M., Marcote, S. L., Durieux, T., Le Berre, D., & Monperrus, M. (2016). Nopol: Automatic repair of conditional statement bugs in java programs. *IEEE Transactions on Software Engineering*, *43*(1), 34-55.

Yilmaz, R., & Yilmaz, F. G. K. (2023). Augmented intelligence in programming learning: Examining student views on the use of ChatGPT for programming learning. *Computers in Human Behavior: Artificial Humans*, *1*(2), 100005.

Zhang, Q., Zhang, T., Zhai, J., Fang, C., Yu, B., Sun, W., & Chen, Z. (2023). A critical review of large language model on software engineering: An example from chatgpt and automated program repair. *arXiv preprint arXiv:2310.08879*.

Zhong, H., & Su, Z. (2015, May). An empirical study on real bug fixes. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering* (Vol. 1, pp. 913-923). IEEE.

Access this Article in Online

Website: www.ijarm.com

Subject: Software Engineering

Quick Response Code

DOI:10.22192/ijamr.2024.11.03.005

How to cite this article:
Hussaini Dan'azumi, Dr. Yakubu Bala Mohammed, and Mahmood Saidu Badara. (2024). Exploring the Use of ChatGPT for Resolving Programming Bugs. Int. J. Adv. Multidiscip. Res. 11(3): 34-44.
DOI: http://dx.doi.org/10.22192/ijamr.2024.12.03.005