EAI EndorsedTransactions
on AI and Robotics
Research Article EAI.EU

The Potential Use of ChatGPT for Debugging and Bug Fixing

Md. Asraful Haque[1,*] and Shuai Li[2]

[1]Department of Computer Engineering, Aligarh Muslim University, Aligarh-202002, India
[2]Faculty of Information Technology & Electrical Engineering, University of Oulu, Finland

Abstract

ChatGPT is a cutting-edge language model that has been making waves in the field of natural language processing. However, its capabilities extend far beyond language-based applications. ChatGPT can also be used as a powerful tool for debugging software code. As software applications become increasingly complex, the need for efficient and accurate debugging tools has become more pressing. ChatGPT's ability to analyze and understand code makes it a promising solution to this challenge. Debugging is a critical part of the software development process. Bugs, or errors in code, can have serious consequences for the functionality and security of software applications. Identifying and fixing bugs can be a time-consuming and labor-intensive process, requiring the expertise of experienced developers. ChatGPT has the potential to streamline this process and make it more accessible to a wider range of developers, regardless of their experience level. In this article, we will explore the capabilities of ChatGPT as a debugging tool, the advantages and limitations of using it, and best practices for integrating it into the software development workflow.

Keywords: ChatGpt, Lamguage Model, Debugger, GPT-3.5.

Received on 24 April 2023, accepted on 03 May 2023, published on 11 May 2023

doi: 10.4108/airo.v2i1.3276

1. Introduction

Debugging is the process of identifying and fixing errors, or bugs, in software code [1]. Bugs can occur for a variety of reasons, including typos, incorrect syntax, incorrect logic, or unexpected inputs. When a bug is present in code, it can cause the program to behave unexpectedly, produce incorrect results, or even crash. Debugging is a critical part of the software development process, as it ensures that software applications are functioning correctly and efficiently [2]. Programmers rely on debugging to diagnose these faults and perform post-mortem analyses [3]. The debugging process typically involves a series of steps, starting with identifying the error and tracing it back to its root cause. Thus, it can be a time-consuming and complex process, often requiring the expertise of experienced developers who are able to identify the root cause of the problem and implement a fix. Sometimes different debugging tools are used to locate the source of the error. Many programmers (as well as organizations) employ software visualisation in an effort to better comprehend the dynamics of programme execution [4][5]. Figure 1 shows the typical debugging process of traditional method. Once the error is identified, developers can then implement a fix to resolve the issue. Coding modifications, adjusting settings or configurations, and implementing other changes to the software are all part of the bug-fixing process [6-8]. Traditional debugging often involves a lot of trial and error, and can be time-consuming and frustrating. An effective debugging and bug fixing process require a combination of technical skills, collaboration, and attention to detail. By implementing best practices and using the right tools and techniques, developers can ensure that their software is error-free and reliable.

*Corresponding author. Email: md_asraf@zhcet.ac.in

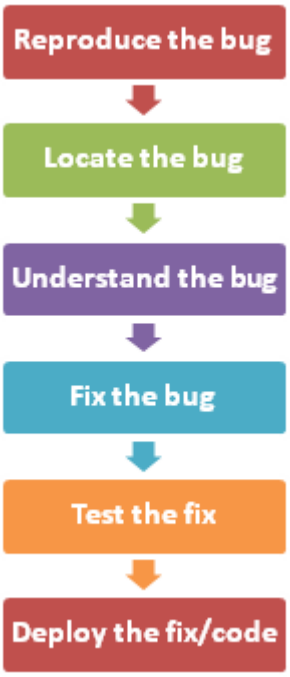

Figure 1. Typical Debugging Process using Traditional Methods

ChatGPT is a state-of-the-art language model based on the GPT-3.5 architecture [9]. Since its debut on November 30, 2022, ChatGPT has become incredibly popular as users experiment with it and become more familiar with its features [10]. It has many unknown capabilities that can revolutionize various industries, from e-commerce to mental health care [11-15]. While these capabilities are still in their early stages, they have the potential to transform the way we live, work, and communicate. ChatGPT has the ability to perform a wide range of tasks, including debugging [16-18]. As a debugger, ChatGPT uses its advanced natural language processing (NLP) capabilities to analyze and identify errors in code. No matter what programming language is used, it can process code written in a number of ways and find problems. It is also capable of understanding the context of the code and can provide suggestions to resolve the error in the code. In this paper, we look into ChatGPT's capabilities as a debugging tool, as well as the benefits and drawbacks of utilising it.

2. How does ChatGPT work as a debugger?

ChatGPT is a natural language processing model, which means it is trained to understand human language. However, it can also be trained on programming languages and the syntax used in software code. When ChatGPT encounters a piece of code, it is able to analyze it and identify potential issues. ChatGPT can detect a wide range of errors in code, including syntax errors, logical errors, and semantic errors. Syntax errors are errors in the way the code is written, such as missing or extra punctuation or incorrect syntax. Logical errors occur when the code does not behave as expected, even though it is written correctly. Semantic errors occur when the code is technically correct, but does not produce the intended results. ChatGPT can work as a debugger in several ways. It can provide assistance with understanding code by answering questions about syntax, function behavior, or code structure. For example, a developer can ask ChatGPT about the definition or usage of a specific function, and ChatGPT can provide a detailed explanation that helps the developer understand how to use the function correctly. It can help with identifying and correcting syntax errors, such as missing brackets or semicolons, by suggesting corrections or pointing out where the error is occurring. ChatGPT can also help with identifying the root cause of issues by clarifying error messages, providing context on the environment in which the code is running, or suggesting potential solutions to a problem.

Figure 2. Debugging Process using ChatGPT

Figure 2 illustrates the general steps that ChatGPT might take to debug a piece of code and fix a bug:

- Input: The developer provides the code to ChatGPT along with a description of the issue they are experiencing.
- Analysis: ChatGPT analyzes the code using natural language processing and machine learning techniques. It identifies potential issues, such as syntax errors or logical inconsistencies, and generates a list of possible solutions.
- Ranking: ChatGPT ranks the possible solutions based on their relevance and likelihood of success. It may consider factors such as the developer's past behavior, code patterns, and the success rates of similar solutions.
- Suggestion: ChatGPT generates a suggestion for how to fix the bug, typically in natural language. The suggestion may include changes to the code itself, or it may recommend changes to the environment or configuration settings.
- Feedback: The developer evaluates the suggestion and provides feedback to ChatGPT. They may accept the suggestion, modify it, or reject it altogether. This feedback is used to improve the accuracy of future suggestions.
- Integration: If the developer accepts the suggestion, they integrate the suggested fix into their codebase. ChatGPT may also provide tools for automating the integration process, such as code refactoring or testing frameworks.

To demonstrate how ChatGPT can assist with debugging and bug fixing, let's consider a simple C-program: Suppose we have a C-program that is intended to calculate the sum of two integers entered by the user (Table 1).

Table 1. C-code to add two integers

```
1   #include <stdio.h>
2   int main()
3   {
4   int num1, num2, sum;
5   printf("enter the first integer: ");
6   scanf("%d", &num1);
7   printf("Enter the second integer: ");
8   scanf("%d", &num2);
9   sum = num1 –  num2;
10  printf("The sum of %d and %d is %d",
    num1, num2, sum);
11  return 0;
12  }
```

Now we can see, there is an error in the code: the line sum = num1 – num2; should be sum = num1 + num2; in order to calculate the correct sum. To identify and fix this error, we could ask ChatGPT for assistance. We might ask:

"Can you help me debug my C-program that is supposed to calculate the sum of two integers, but is giving an incorrect result?"

ChatGPT may respond with some suggestions for debugging and fixing the error, such as:

(i) Check that the variables are initialized properly and are of the correct data type.
(ii) Use print statements to check the values of the variables at different stages of the program.
(iii) Verify that the mathematical operation being performed is correct.

Following these suggestions, we would be able to identify and fix the error in the code by changing sum = num1 – num2; to sum = num1 + num2; and the program would be able to correctly calculate the sum of two integers.

While ChatGPT is not able to directly debug or fix code, it can provide guidance and suggestions to help users identify and resolve errors in their code, such as through suggestions for debugging and verifying that mathematical operations are correct. Consider a Python code as another example (Table 2).

Table 2. Python code to divide two numbers

```
1   def divide(a, b):
2   try:
3   return a / b
4   except ZeroDivisionError:
5   print("error: cannot divide by zero.")
```

This code defines a function ‘divide’ that takes two arguments ‘a’ and ‘b’ and returns their division, unless ‘b’ is zero, in which case it prints an error message. Let's assume that a developer encounters a bug while testing this code – if ‘b’ is zero, the function does not return anything, causing unexpected behavior in the calling code. The developer can use ChatGPT to help diagnose and fix this bug. To do this, the developer can provide the code to ChatGPT along with a brief description of the issue. For example:

Table 3. ChatGPT fixing ZeroDivisonError

```
1   def divide(a, b):
2   try:
3   return a / b
4   except ZeroDivisionError:
5   print("error: cannot divide by zero.")
```

Description:

The divide function does not return anything when b is zero, causing unexpected behavior in the calling code. How can I fix this?

ChatGPT can then analyze the code and provide suggestions for a fix in natural language. In this case, it may suggest modifying the 'except' block to raise an exception instead of printing the error message. The developer can then evaluate this suggestion and determine if it is a suitable fix for the issue.

3. Advantages of using ChatGPT as a debugger

ChatGPT has the ability to perform deep analysis of code [19]. It uses machine learning to analyze code and provide suggestions for debugging and bug fixing. There are several advantages of using ChatGPT as a debugger:

3.1. Greater Efficiency and Accuracy

Traditional debugging can be a time-consuming process, particularly when trying to track down elusive bugs. ChatGPT can analyze code much faster than human developers and can identify errors in code with a high degree of accuracy. Its ability to understand and analyze natural language could be leveraged to automate the debugging process. By providing access to code repositories and error logs, ChatGPT could identify and fix bugs automatically, saving programmers time and effort. ChatGPT could be integrated with augmented reality devices to provide real-time feedback and suggestions for debugging code. This would allow programmers to visualize their code and see the impact of changes in real-time, improving the debugging process and reducing the risk of introducing new bugs. By improving the efficiency and accuracy of the debugging process, ChatGPT can save developers time and reduce the overall cost of software development.

3.2. Improved Code Quality

ChatGPT can also provide suggestions for improving code quality. It can identify code that is difficult to maintain, inefficient or redundant, and suggest improvements that can make the code more readable and easier to maintain [20]. It can provide suggestions for the most appropriate coding practices, including best practices for documentation, code structure, and variable naming conventions. This helps developers to write clean, maintainable code from the outset, reducing the likelihood of errors occurring in the first place.

3.3. Chat-based Debugging and Increased Accessibility

An important factor in an efficient debugging and bug fixing environment is collaboration. In complex software projects, multiple developers may be working on different aspects of the codebase simultaneously. Effective communication and collaboration are essential to ensure that errors are identified and resolved successfully and quickly. As ChatGPT is a language model that is designed to converse with humans in a natural language, it could be used as a chatbot to assist programmers in debugging their code. By conversing with the chatbot, programmers could receive immediate feedback and suggestions for how to solve their programming problems. This makes ChatGPT an ideal tool for collaboration. It can communicate with other team members, including non-technical stakeholders, to explain technical issues and suggest solutions. Thus, the team members with varying levels of technical expertise or even inexperienced developers can use ChatGPT to work together to resolve issues, reducing the need for specialized expertise in the debugging process.

3.4. Continuous Learning

ChatGPT can be trained on new programming languages and syntax, allowing it to continuously learn and improve its ability to identify and fix errors in code. This means that it can learn from previous debugging sessions. It can remember the errors and solutions that were used to fix them, and use this information to provide more accurate suggestions in future debugging sessions. ChatGPT becomes more effective over time, as it accumulates more data and learns from its previous experiences.

3.4. Debugging as a Service

With the growing popularity of cloud-based services, ChatGPT could be used to offer debugging as a service to programmers. This would provide a scalable and cost-effective solution for companies and developers who lack the expertise to debug their own code. This can make software development more accessible to smaller teams or organizations with limited resources.

The advantages of using ChatGPT as a debugger make it a promising tool for improving the efficiency and accuracy of the software development process.

4. Limitations of using ChatGPT as a debugger

While ChatGPT has many advantages as a debugger, there are also several limitations that developers should be aware of:

4.1. Limited Understanding of Context

While ChatGPT is able to analyze and understand code, it does not have a deep understanding of the broader context in which the code is being used. This means it may not have the same level of context awareness as a human programmer. This could lead to misunderstandings or incorrect assumptions about the code that needs to be debugged. It may suggest fixes that are not appropriate for the specific use case or application.

4.2. Dependence on Training Data and Limited Knowledge Base

ChatGPT recognizes patterns in the code and detects deviations from those patterns by comparing them with training data. Its ability to identify and fix errors is dependent on the quality and relevance of its training data. If the training data is outdated or inaccurate or if the training data does not include all possible errors or solutions, ChatGPT may not be able to accurately identify or suggest fixes for errors. ChatGPT is not a technical expert and does not have the same level of understanding of programming languages and debugging techniques as a human programmer or a dedicated debugging tool. Therefore, it may struggle to provide accurate and useful advice when it comes to debugging code.

4.3. Inability to Address Complex Errors

Debugging often requires an in-depth understanding of programming concepts, architecture, and tools, as well as a deep understanding of the codebase being debugged. ChatGPT may not have the same level of technical expertise required to address complex debugging issues. It relies on the inputs it receives and its pre-existing knowledge base to generate responses, which may not be sufficient for addressing complex problems. As a result, ChatGPT could find it difficult to handle complicated errors that demand an in-depth comprehension of the code and the larger context in which it is being utilised. In these cases, human expertise may be required to identify and fix the error.

4.4. Lack of Real-Time Interaction

Debugging often requires real-time interaction with the code and the ability to step through it line by line to identify issues. ChatGPT does not have the ability to execute code directly. This means that it cannot identify runtime errors or perform other real-time tasks that require direct access to the code. ChatGPT is not designed for this type of real-time interaction and may struggle to keep up with the pace of debugging activities.

It is important to note that ChatGPT can be a useful tool for providing general guidance and advice on programming and debugging, it is not designed or optimized for use as a debugger. It doesn't have an in-depth understanding of programming concepts and techniques that are often required for a dedicated debugger. Developers should exercise caution when implementing its suggestions and ensure that they have a deep understanding of the broader context and requirements of the application. ChatGPT should be viewed as a complementary tool to human expertise, rather than a replacement for it.

5. Best practices for using ChatGPT as a debugger

Debuggers are specialized tools designed specifically for debugging code, and they offer a wide range of functionalities that go beyond what ChatGPT can offer. For example, debuggers allow developers to set breakpoints, inspect the values of variables, step through code, and examine the call stack, all of which are critical for identifying and fixing complex bugs. Debuggers also provide real-time feedback, making it easier for developers to catch errors as they occur. ChatGPT, on the other hand, relies on previously logged debugging data and may not be able to provide real-time feedback or offer the same level of functionality as traditional debuggers. It also has limited access to the codebase, which can affect its ability to provide accurate suggestions for fixes. Therefore, while ChatGPT has shown promising capabilities as an AI-based debugger and bug fixer, there are several limitations that need to be considered. Developers should use ChatGPT as a complementary tool alongside traditional debugging techniques to improve the efficiency and effectiveness of their software development process. Here are some best practices for using ChatGPT as a debugger:

(i) Understand the Limitations: It is important to understand the limitations of ChatGPT as a debugger. While it can identify and suggest fixes for many common errors, it may struggle with more complex issues that require human expertise.

(ii) Use Multiple Sources: Don't rely solely on ChatGPT for debugging. Use multiple sources, including other debugging tools and human expertise, to verify and validate ChatGPT's suggestions.

(iii) Provide Context: When using ChatGPT as a debugger, provide as much context as possible. This includes the specific error message, the code that produced the error, and any relevant logs or output. The more context you provide, the more accurate ChatGPT's suggestions will be.

(iv) Train ChatGPT: Consider training ChatGPT on your specific codebase or project. This can improve its ability to identify and suggest fixes for errors that are unique to your project.

(v) Verify Suggestions: Always verify ChatGPT's suggestions before implementing them. This includes checking that the suggested fix is appropriate for the specific use case and that it aligns with the broader objectives of the project.

(vi) Provide Feedback: Finally, provide feedback to ChatGPT. This includes letting it know when its suggestions were helpful and when they were not. This can help improve its accuracy and relevance for future debugging sessions.

By following these best practices, we can improve the accuracy and effectiveness of ChatGPT as a debugger.

6. Future developments

As ChatGPT continues to evolve and improve, we can expect to see even more innovative applications and use cases emerge [21][22]. ChatGPT can be useful as an AI-based debugger that can provide suggestions in real-time and automate debugging tasks. However, many automated debugging tools are already available in the market (i.e. ControlFlag, LambdaTest etc). An automatic debugger is a software program that assists in the debugging process by automatically detecting and diagnosing issues in code. Automatic debuggers typically use a combination of static and dynamic analysis techniques to identify potential errors or bugs in code, and can even suggest possible solutions to the problem. One of the key benefits of automatic debuggers is that they can often identify issues that might be difficult or time-consuming to find manually, such as runtime errors or logic errors. ChatGPT and other automatic debuggers have different strengths and weaknesses. ChatGPT is more flexible, has the ability to learn from previous debugging sessions, and can communicate in natural language, making it easier to collaborate with team members. However, it may have some errors due to limitations in its language model and may have limited integration with development tools. Automatic debuggers, on the other hand, have high accuracy due to pre-defined rules and algorithms. They may not provide deeper insights into code quality and may have limited collaborative capabilities. Ultimately, the choice between ChatGPT and automatic debuggers will depend on the specific needs of the development team and the software project. A general comparison between ChatGPT and other automated debugging tools is shown in Table 4. It is important to note that ChatGPT as a debugging tool is still in its early stages and has the potential for significant future developments. Here are some potential areas for future development:

(i) Improved Accuracy: One of the primary areas for improvement is in the accuracy of ChatGPT's suggestions. As the model is trained on more and more data, it will become better at identifying and suggesting fixes for errors.

(ii) More Complex Errors: ChatGPT currently struggles with more complex errors that require human expertise. Future developments may enable ChatGPT to identify and suggest fixes for these types of errors.

(iii) Integration with IDEs: Integrating ChatGPT with Integrated Development Environments (IDEs) could make it more convenient for developers to use. By embedding ChatGPT directly into the IDE, developers could get real-time suggestions and feedback on their code.

(iv) Custom Training: ChatGPT's accuracy could be improved by custom training on specific codebases or projects. This would enable the model to better understand the context and specificities of each project.

(v) Collaboration with Human Debuggers: ChatGPT could collaborate with human debuggers to improve its accuracy and relevance. By working together, human debuggers could provide feedback and corrections to ChatGPT, while ChatGPT could help automate the debugging process.

(vi) Integration with Automated Testing: ChatGPT could also be integrated with automated testing tools to help identify and fix errors as they occur during the testing phase. This could significantly reduce the time and effort required for manual debugging.

ChatGPT as a debugging tool has significant potential for future development. By addressing the above issues, ChatGPT could become an even more powerful debugging tool in the future.

Table 4. Comparison between ChatGPT and Automated debugging tool

Criteria	ChatGPT	Automated debugging tool
Type of Debugger	AI-based debugger that uses natural language processing	Automated debugger that uses pre-defined rules and algorithms
Ability to Learn	Can learn from previous debugging sessions	Does not learn from previous debugging sessions
Flexibility	Can process code written in any programming language	Limited to specific programming languages
Accuracy	May have some errors due to limitations in language model	High accuracy due to pre-defined rules and algorithms
Context Awareness	Can understand the context of the code and provide context- specific suggestions	Limited context awareness and may miss some errors due to lack of context
Collaborative	Can communicate in natural language with team members	Limited collaborative capabilities
Speed	Can provide suggestions in real-time and automate debugging tasks	Can provide suggestions quickly, but may require manual implementation of fixes
Skill Level Required	No specialized technical skills required	Requires technical knowledge and expertise
Overall Performance	Can provide valuable insights and suggestions for improving code quality	Can quickly identify errors, but may not be able to provide deeper insights into code quality

6. Conclusion

ChatGPT is a powerful language model that has gained widespread recognition for its ability to generate high-quality text, engage in natural language conversations, and perform a variety of language-related tasks. However, there are many lesser-known capabilities of ChatGPT that are equally impressive and deserve greater attention. It is able to identify errors by analyzing the code and comparing it to its training data. Despite the fact that ChatGPT has the potential to be a useful tool for debugging and bug repair, it should be used with caution and as part of a thorough strategy to software development. Its ability to learn from previous debugging sessions and provide natural language suggestions can be valuable for improving code quality, but it also has several limitations, including limited domain knowledge, context awareness, and integration with development tools. While ChatGPT can help developers save time and effort by automating certain aspects of debugging and bug fixing, it should not be relied upon solely to catch and fix all errors. Human review and testing remain critical components of the software development process, and developers should use ChatGPT's suggestions as a starting point, rather than blindly implementing them without careful consideration and testing. Overall, ChatGPT should be seen as a tool to augment and enhance the skills and expertise of human developers, rather than as a replacement for them. By using ChatGPT in conjunction with other debugging and bug fixing techniques, developers can improve the efficiency and effectiveness of their software development process and deliver higher-quality software to their users.

References

[1] Wotawa F, Nica M, Moraru I. Automated debugging based on a constraint model of the program and a test case. The Journal of Logic and Algebraic Programming, Volume 81, Issue 4, 2012, Pages 390-407.

[2] Srivastva S, Dhir S. Debugging approaches on various software processing levels. Int. conference of Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 2017, pp. 302-306.
[3] Haque MA. Problems in Aspect Oriented Design: Facts and Thoughts. Int. Journal of Computer Science Issues, Vol. 8, Issue 2, pages 552-556, March 2011.
[4] Steven PR. Visual representations of executing programs. Journal of Visual Languages and Computing 18(2) pp. 126-148 (2007).
[5] Bas C. et al. A systematic survey of program comprehension through dynamic analysis. Technical Report TUDSERG-2008-033, Delft University of Technology, 2008.
[6] McCauley R et al. Debugging: a review of the literature from an educational perspective, Computer Science Education, 18:2, 67-92, 2008.
[7] Andrew J. et al. Debugging reinvented: asking and answering why and why not questions about program behaviour. Int. Conference on Software Engineering 2008, pp. 301-310.
[8] Chmiel R, Loui MC. Debugging: from novice to expert. ACM SIGCSE Bulletin 36(1): 17- 21, 2004.
[9] Introducing ChatGPT: https://openai.com/blog/chatgpt.
[10] Haque MA. A Brief Analysis of 'ChatGPT' – A Revolutionary Tool Designed by OpenAI. EAI Endorsed Transactions on AI and Robotics, vol. 1, no. 1, p. e15, Mar. 2023.
[11] George AS, George AH. A Review of ChatGPT AI's Impact on Several Business Sectors. Partners Universal International Innovation Journal, 1(1), pp.9-23, 2023.
[12] Frederico GF. ChatGPT in Supply Chains: Initial Evidence of Applications and Potential Research Agenda. Logistics. 2023; 7(2):26.
[13] Wen J, Wang W. The future of ChatGPT in academic research and publishing: A commentary for clinical and translational medicine. Clin Transl Med. 2023; 13:e1207.
[14] Sallam M. ChatGPT Utility in Healthcare Education, Research, and Practice: Systematic Review on the Promising Perspectives and Valid Concerns. Healthcare. 2023; 11(6):887.
[15] PP Ray. ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. Internet of Things and Cyber-Physical Systems, Volume 3, 2023, Pages 121-154.
[16] Hassani H, Silva ES. The Role of ChatGPT in Data Science: How AI-Assisted Conversational Interfaces Are Revolutionizing the Field. Big Data Cogn. Comput. 2023, 7(2), 62.
[17] Thomas C. ChatGPT becomes ChatRepair to automate bug fixing for less. The Register, RSA Conference, April-2023. https://www.theregister.com/2023/04/05/chatrepair_automates_bug_hunting/.
[18] A. Haleem, M. Javaid and RP Singh, "An era of ChatGPT as a significant futuristic support tool: A study on features, abilities, and challenges", BenchCouncil Transactions on Benchmarks, Standards and Evaluations, vol. 2 (4), 2023.
[19] Cao J et al. A study on Prompt Design, Advantages and Limitations of ChatGPT for Deep Learning Program Repair. https://arxiv.org/pdf/2304.08191.pdf, 2023.
[20] Taecharungroj V. "What Can ChatGPT Do?" Analyzing Early Reactions to the Innovative AI Chatbot on Twitter. Big Data Cogn. Comput. 2023, 7, 35.
[21] Mathew A. Is Artificial Intelligence a World Changer? A Case Study of OpenAI's Chat GPT. Recent Progress in Science and Technology, Vol. 5, pp.35-42, 2023.
[22] Hughes A. ChatGPT: Everything you need to know about OpenAI's GPT-3 tool. BBC Science Focus Magazine, 2023.