https://doi.org/10.36315/2023v2end126
Education and New Developments 2023

ARTIFICIAL INTELLIGENCE CHATBOTS – A HELP OR HINDRANCE TO COMPUTER SCIENCE EDUCATION

Paul Sage
School of Electronics, Electrical Engineering and Computer Science,
Queen's University Belfast (United Kingdom)

Abstract

Recent developments in artificial intelligence (AI) chatbot systems have gained significant coverage over the past few months across all subject disciplines and educational levels. Emergent technologies such as Chat GPT from Microsoft and BARD from Google have demonstrated extraordinary use of machine learning to enable detailed responses with high-utility for simple text-based queries. Students at all levels are broadly aware of the significance of such systems in providing an apparently easy route to homework solutions, with associated concerns from educators. While many consider such systems to be a threat to educational practices, others have embraced the technology, exploring these capabilities to support and enhance learning and development.
The work detailed in this paper considers the use of Chat GPT in suggesting solutions to simple programming problems, typically used when leaning to develop software. Problem specifications from current programming assignments in year 1 of an undergraduate computing degree program are considered in the context of this system, over a range of input fragments and compared against utility of output (functionality and correctness of generated code). Responses generated from Chat GPT for a problem specification are considered alongside current student work and submitted to a blind assessment process.
Results indicate that for simple problems, a significant proportion of code generated through Chat GP produces a fairly high utility, although amendments are required in all cases to enable testing. In many cases, text directly lifted from a problem specification provided enough material for Chat GPT to generate a reasonable response, although increased complexity resulted in reduced utility.
The paper provides an overview analysis of initial experimentation and results, focusing specifically on how such systems could potentially benefit the novice programmer.

Keywords: *Generative artificial intelligence, programming.*

1. Introduction

The end of 2022 saw the emergence of generative artificial intelligence in the form of a new generation of tools capable of providing seemingly accurate, human-like responses to standard text-based queries. While chatbots have been around for some time, these new tools provide a layer above that of a simple 'google search' for information on the internet, moreover, providing bespoke responses to specific queries. ChatGPT (Brown et al., 2020) backed by Microsoft has gained significant traction through recent media exposure, but others exist, such as Bard (Mazare et al., 2018) from Google, with, undoubtedly, more to follow.

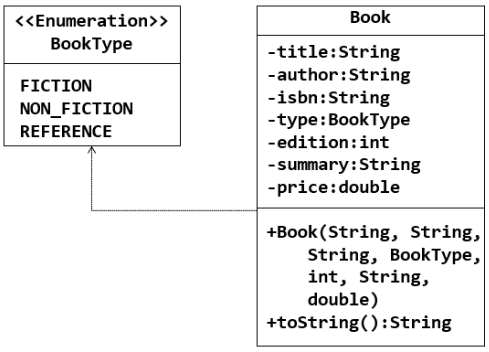
These systems have rapidly become of significant interest to educators as their ease of use, availability, and ability to generate accurate and useful responses when asked to generate results for student assignment specifications, have made it possible for students to circumvent the usual assessment of learning outcomes. There are many recent examples of using ChatGPT to generate full essays in complex topics in any style requested by the user, with potentially boundless possibilities for misuse. An understandable response would be to push back against this, perhaps looking to ban (or stymie) the use of generative AI within education assessment. While controlled assessments offer some solution towards this, it is impossible to prevent access to these resources when considering continuous assessment. The genie is indeed out of the bottle.

One eye-catching example of the use of generative AI is its application to the generation of computer programs. ChatGPT can generate code in any chosen language from simple text-based queries. Consequently, the purpose of this paper is to examine the utility of generated output for programming problems, typically undertaken by a year 1 cohort as part of an undergraduate course of study in Computer Science. Furthermore, this work looks beyond the knee-jerk reaction to systems such as ChatGPT, in suggesting how they may be put to effective use in Computer Science education, with some recommendations towards learning support and sustainable assessment.

2. Experimentation

Under consideration is the use of ChatGPT for generating code for year 1 programming problems. By their nature, these problems assess the first steps in learning to program, covering fundamental aspects of data management and code structure using specifications modelling real-world entities. Such specifications, expressed through a Universal Modelling Language (UML) (Rumbaugh, Jacobson, & Booch, 2005) form the basis of formative and summative assessment problems. Figure 1 illustrates the description of an object to represent a book.

Figure 1. UML for a Book Object.



A typical problem using this definition would be to (for example) generate Java code to represent a Book object from a given UML specification. While at time of writing, ChatGPT (version 3) does not offer the opportunity for the student to input images (such as Figure 1), this is readily translated to text form, yielding a query as follows:

"**Add a class called Book to the part01 package – add instance variables and constructor as described in figure 1. The constructor should be provided initial values for title, author, isbn, type, edition, summary and price**."

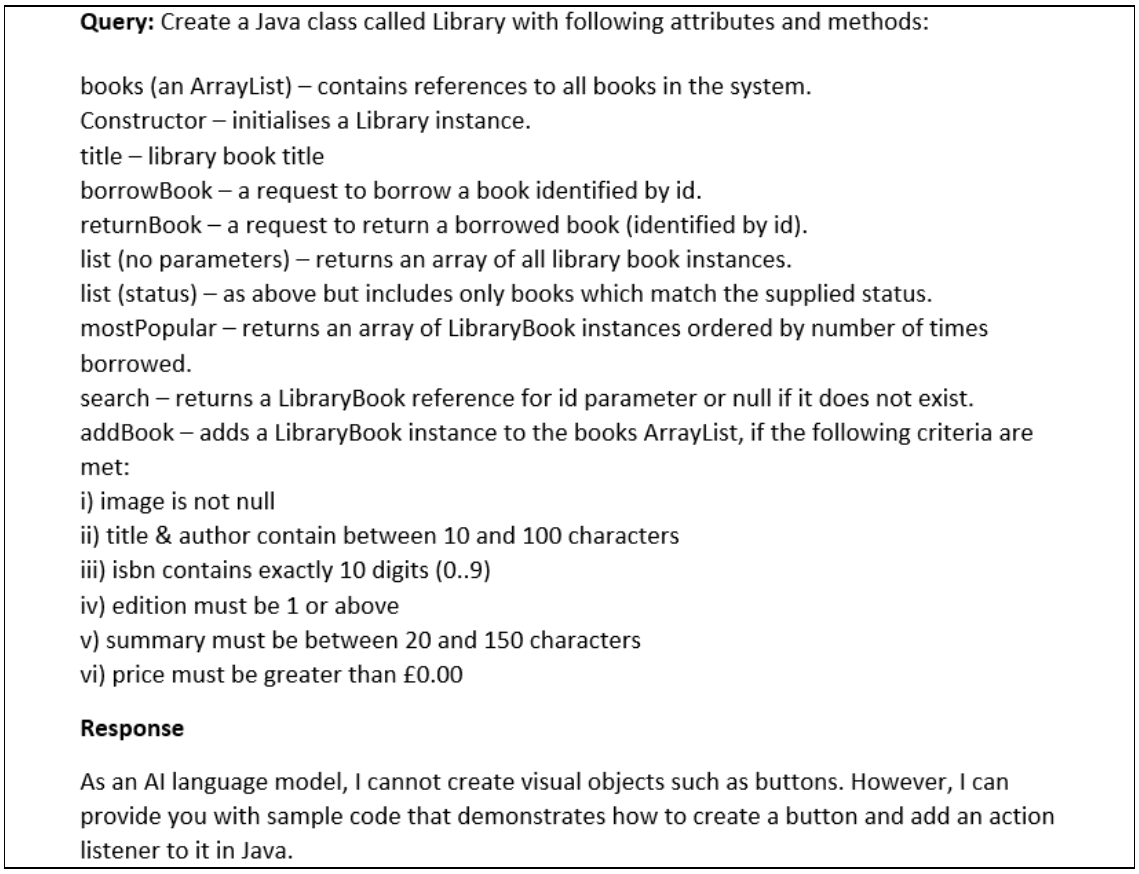
When offered to ChatGPT, the response produced perfect code. Even though references to information in 'Figure 1' was not visible to the system, it was able to infer what was required, due to the prescriptive nature of the query and its understanding of what a book is (or means). So, anything that is (or represents) a commonly used term or entity for the most part, is easily managed by ChatGPT and extrapolating this outcome to cover a full assessment of fundamental programming concepts does not present a significant challenge in the main.

This was put to the test in a selective decomposition of a year 1 programming assessment, covering fundamental structures in object-based programming (McCarthy, 1995), taken after the first five weeks of study. The generated output was then assembled in the form of a student submission and subsequently submitted for blind assessment. The result of assessment returned a 100% utility and score for the submission. This is not such a concern as this assessment was taken within a controlled lab environment with no access to external resources.

However, careful decomposition of a problem specification is required to obtain the most effective response as ChatGPT does not always get things correct and utility largely depends on the quality of input. Moreover, the use of bespoke or non-standard descriptions of components within a problem specification does not always produce a useful response and often leads ChatGPT to produce a result clearly out of context from what has been asked. Figure 2 illustrates a slightly less decomposed query, taken directly from a problem specification. In this case, the query response was inappropriate and unusable in the context of the problem, resulting in code for components associated with graphical user interfaces, which were not mentioned or required in the problem specification.

Since the utility of generated output is related to the quality of input through effective problem decomposition, it’s clear that some fundamental understanding of the problem domain is required on the part of the (student) user. Although the future may indeed yield a more useful result for poorer quality input, understanding the basics can both benefit the student and the educator in coping with or possibly promoting the use of generative AI as an integral part of the learning process.

Figure 2. A More Complex Query.



Query: Create a Java class called Library with following attributes and methods:

books (an ArrayList) – contains references to all books in the system.
Constructor – initialises a Library instance.
title – library book title
borrowBook – a request to borrow a book identified by id.
returnBook – a request to return a borrowed book (identified by id).
list (no parameters) – returns an array of all library book instances.
list (status) – as above but includes only books which match the supplied status.
mostPopular – returns an array of LibraryBook instances ordered by number of times borrowed.
search – returns a LibraryBook reference for id parameter or null if it does not exist.
addBook – adds a LibraryBook instance to the books ArrayList, if the following criteria are met:
i) image is not null
ii) title & author contain between 10 and 100 characters
iii) isbn contains exactly 10 digits (0..9)
iv) edition must be 1 or above
v) summary must be between 20 and 150 characters
vi) price must be greater than £0.00

Response

As an AI language model, I cannot create visual objects such as buttons. However, I can provide you with sample code that demonstrates how to create a button and add an action listener to it in Java.

3. Analysis

Generative AI systems are now a reality and will form a part of future learning and assessment. However, the threat does not necessarily come from how students may make use of these tools to cheat on assessments, more so from a misunderstanding of the role such systems can have, as an integral part of the learning process. Clarification is needed on when it’s appropriate to prevent their use and when to promote and encourage usage. Space limits a full discussion on the possible impact of generative AI on computer science education but some points to note for teaching fundamental programming concepts are provided below:

1. ***Generative AI can be a major benefit in supporting larger class groups***: Over the past number of years, computing-related degree programs have become increasingly popular, leading to very large class sizes, particularly in year 1 modules. This has led to poor staff student ratios with limited opportunities for one-to-one learning. For fundamental programming concepts, prescriptive queries (for ChatGPT) typically produce accurate code responses, supplemented with contextual descriptions and example usage, bespoke to the student. This represents an invaluable lab resource, which could also be (potentially) used to inform the educator of student progress in real-time, enabling the identification of areas of weakness and recommendations for further study.
2. ***Summative assessment of programming fundamentals should take place in a controlled environment***: while the use of ChatGPT (for example) should be encouraged to assist the student with their conceptual understanding of course material and formative assessments, the summative assessment of the practical application of fundamentals should be in the context of controlled (lab-based) assessments, without the use of generative AI.

3. ***Continuous assessment (summative) should not be discontinued***: Throwing a complete problem specification at ChatGPT yields a poor response – some decomposition is required. Unless the specification is prescriptive, such decomposition requires application of fundamental knowledge. When problem specifications form an integral part of learning material, for example within project-based learning, additional work (and understanding) is required on the part of the student to obtain an effective response from generative AI. Furthermore, where the assessment involves the design and development of bespoke components, generative AI is less able to produce useable code but does instead offer a developmental route-map, which would form an acceptable support scaffold for students.
4. ***Prescriptive problem specifications should not necessarily be avoided***: through suitable information, students should be made aware of the implications of using generative AI to short-circuit the assessment process. Using prescriptive specifications typically results in standard responses from ChatGPT, which are relatively easy to spot and manage.

4. Conclusions

Generative AI is a relatively new topic, but it has already been acknowledged that it will have a significant impact on all aspects of life, with education a specific concern. Students are already well aware of what generative AI is capable of, with further (and certain) developments furthering concerns for the educator. By and large, generative AI systems can be easily purposed to generate (for example) computer code with limited effort. However, opportunities exist to enhance the student learning experience, through integration of generative AI into the curriculum but this will depend on necessary adjustments to both how assessments are specified and how learning outcomes are assessed.

References

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Amodei, D. (2020). Language Models are Few-Shot Learners. In D. Precup & Y. W. Teh (Eds.), *Proceedings of the 37th International Conference on Machine Learning (ICML)*, Vol. 119, pp. 140-151.

Mazare, L., Liu, Y., Liu, M., Cao, Y., Zettlemoyer, L., & Lapata, M. (2018). Generating Responses with a Specific Emotion in Dialog. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 527-536.

McCarthy, J. (1995). Object-Based Programming: A Comprehensive Review. *Journal of Object-Oriented Programming, 8*(3), 45-62.

Rumbaugh, J., Jacobson, I., & Booch, G. (2005). *The Unified Modeling Language Reference Manual* (2nd ed.). Addison-Wesley Professional.