TOWARDS EFFECTIVE AUTOMATED GRADING IN CS1: A COMPARISON BETWEEN GenAI AND THE IN-HOUSE GRADING TOOL

Pieter Joubert, Wendy Yanez-Pazmino, and Jacqui Chetty
University of Birmingham
United Kingdom

Abstract

The adoption of Generative Artificial Intelligence (GenAI) in automating assessment has become increasingly popular in introductory computer science (CS1) modules, especially for large student cohorts. This study evaluates the effectiveness of a GenAI-based grading tool built using OpenAI API, in comparison to a traditional automated grading system. We conducted the study on formative assessments submitted by first-year students in the Object-Oriented Programming (OOP) module, which was graded by both systems. The findings reveal that the grades generated by the GenAI were as accurate as those produced by the automated grading system. These results suggest that the integration of GenAI into the grading process for formative assessments can optimise the marking and grading for educators and potentially improve student learning.

Introduction

Artificial intelligence (AI) is a distinctive research field within computer science (CS) (Crompton & Burke, 2023) and its recent advancements, particularly in Generative Artificial Intelligence (GenAI) are reshaping higher education (HE). In HE, the inclusion of AI and GenAI is becoming increasingly popular. This trend has heated intense debate surrounding the advantages, disadvantages, threats and opportunities that AI creates. Without robust policies, ethical frameworks, and collaborative guidance, the adoption of AI in HE may lead to unintended consequences (Bond et al., 2024). Despite a steady rise in research on GenAI in education over the last 5 years, key gaps remain, especially around assessment and grading, where academic integrity concerns are increasing (Tobler, 2024).

Studies show that GenAI is increasingly used for tasks such as grading essays, evaluating free-text responses, and analysing cognitive engagement (Crompton & Burke, 2023). However, traditional coursework assessment and grading are under threat, encouraging educators to rethink how assessments are designed and evaluated (Chan, 2023; Raman & Kumar, 2022). Additionally, these should include the use of GenAI as Rudolph et al. (2023) advise against the idea of policing

students or focusing on academic misconduct for using GenAI. To remedy the current need for evaluating grading methods and providing grading tools that are reliable within CS HE, in this study, a comparison was drawn between the designed automated grading system and a GenAI grading tool.

This study addresses the current need for reliable, scalable tools in computer science education by comparing traditional automated grading systems with a GenAI grading tool. In particular, the study investigates the accuracy and efficacy of the two marking methods when assessing programming assessments. The main contributions of this work are: (a) a GenAI grading tool using different prompts that can be used by lecturers in CS1 modules to automatically grade formative assessments and (b) an empirical comparison of GenAI and automated grading systems in terms of accuracy and performance.

Related Work

Automated grading systems and tools for programming assessments have evolved significantly over the past decade. Traditional approaches often rely on static code analysis or unit testing to evaluate the correctness and functionality of code (Ala-Mutka et al., 2004; Rahman & Nordin, 2007). However, these methods do not fully capture the reliability, efficiency, and complexity of the code, which are critical in educational settings (Van Verth, 1985). Other approaches, like static analysis, are becoming popular among automated testing techniques (Antonucci et al., 2015), as they mainly consider the source code and its abstraction representation, resulting in a fairer evaluation. Similarly, recent work has been done on the use of big data to collect data from programming assessments to identify behavioural patterns and learning flaws, but there is still a large amount of wasted data and tools that cannot fully capture code changes (Antonucci et al., 2015).

Recently, there has been a proliferation of GenAI tools, specifically, Generative Pre-Trained Transformer (GPT-)3 being labelled by *The Economist* as "eerily human-like" and ChatGPT seen as "scary good, crazy-fun" (Kantrowitz, 2022) for automated programming assignment grading. Both GPT-3 and ChatGPT are owned by OpenAI, an organisation that has transformed from a non-profit to a for-profit corporation. Research indicates that ChatGPT can be used for automatic code checking, enabling teaching and grading large groups of students without burdening teacher times (Bang et al., 2023; Jukiewicz, 2024; Mekterovic & Brkic, 2017; Rahman & Nordin, 2007; Rahman & Watanobe, 2023; Raman & Kumar, 2022; Ullah et al., 2018). However, the effectiveness of using GenAI to grade formative assessments for large cohorts, compared to traditional unit test–based automated tools, remains underexplored.

Methods

Description of the Module

This study was conducted within an introductory Object-Oriented Programming (OOP) module offered in the autumn term to both B.Sc. and M.Sc. Computer Science students. The module provides foundational programming skills, covering topics such as control structures, data structures, classes and objects, inheritance, and file handling. Although the module includes both formative and summative assessments, this study focuses on a single formative task, which required students to implement and evaluate their code using lecturer-designed unit tests. Students were expected to verify test outcomes prior to submission. The module is supported by weekly two-hour lectures, two-hour lab sessions, lecturer office hours, and additional one-to-one drop-in sessions.

Motivation for the Project

Over the past four years, assessments in the OOP programming module have been graded using standardised, test-case-based automated grading systems, primarily through unit testing. Lecturers design the unit tests, which are validated by teaching assistants (TAs) to ensure accuracy, reliability, and alignment with the assessment brief. Although the development of assessments and implementation of unit tests is time-intensive, automated grading significantly reduces marking time and promotes fairness by applying consistent evaluation criteria. Students are provided with unit tests to validate their code prior to submission, while an extended set of tests is used for final grading. The system assigns marks mainly based on test outcomes, requiring submissions to be valid, compilable Java code. However, this binary approach, which can be rigid, often results in full marks or zero, limiting edge assessment cases and occasionally necessitating additional moderation.

Data Collection and Analysis

As the purpose of the study was to determine whether GenAI grades assessments as accurately as the automated grading system given the large cohort of students, quantitative research was the most suitable methodology to adopt.

Assessment Structure

A formative programming assessment was designed to evaluate the creation of a single Java class that makes use of methods, constructors, getters, and setters. Students were provided with instructions and a Maven template project as a resource to create and develop the assessment. This project included a selection of unit tests that students could make use of to test their work before submission. It

also included empty Java classes that students needed to update to complete the assessment. The students were given a week to complete the formative assessment and were required to submit the entire project folder to allow the automated grading system to grade it.

Automated Grading

For this module, an automated marking system based on "Junit" tests was used to evaluate students' submitted code, as a project, and to provide a grade. The automated marking system required a working project that could be compiled to run the designed unit tests. The automated marking system also provided limited feedback in the form of a breakdown of the test results in a human-readable format, as shown in Figure 1.

Figure 1

Example Feedback from the Automated Grading System

```
E0Feedback for: xxxxxxxxxx
Marks:(TestName Mark)
    add 1
    multiply 1
    subtraction 1
    division 1
    circle area 0
    sphereVolume      1
Total [ %]:   83.0
Problems found before running tests:
```

GenAI grading

To perform the GenAI grading, a Python script was used to access the OpenAI API, using the "gpt-3.5-turbo-0125" model. Firstly, a prompt without specific assessment information, shown in Figure 2, was run on both assessments. The instructions given to the GenAI grading tool, seen in Figure 2, produced a result in a JSON format to allow for easy analysis.

Figure 2

Prompt 1 Used to Mark Formative Assessments 1 and 2

```
You are a precise markings assistant designed to mark first year java programs and designed to output JSON. In the JSON name each method with its exact method signature and mark each method out of 1. Do not give half marks.
```

Secondly, the specific assessment instructions for each formative assessment (shown in Figure 3 for formative assessment 1 and Figure 4 for formative

assessment 2) were added to the prompt seen in Figure 2 above. All these prompts were run in conjunction with a user prompt to instruct the GenAI grading tool.

Figure 3

Prompt 2 Used to Mark Formative Assessment 1

You are a precise markings assistant designed to mark first year java programs and designed to mark first year java programs and designed to output JSON. In the JSON name each method with its exact method signature and mark each method out of 1. Do not give half marks.
Base the marks on the following question:
Complete the class named BankAccount, which has 4 attributes: clientName, clientID, accountBalance and a boolean to check whether the account is closed or not.
The class includes a constructor with 3 parameters 4 getters (1 for each attribute) and the following methods:
deposit() method: the balance increases with the depositAmount;
withdraw() method: the balance decreases with the withdrawalAmount;
transferTo( ) method: the amount is transferred from the current account to another account (make use of the methods to update both balances accordingly);
closeAccount() method: upon closing an account, the balance should be set to zero
The class and method signatures are provided in the template file BankAccount.java
All methods are tested in BankAccount.java
Please follow the submission instructions on Canvas.

Figure 4

Prompt 2 Used to Mark Formative Assessment 2

You are a precise markings assistant designed to mark first year java programs and designed to output JSON. In the JSON name each method with its exact method signature and mark each method out of 1. Do not give half marks. Base the marks on the following question: Complete the class named BillingManager, which includes a default constructor with VAT = 20%, a second constructor that takes in the VAT parameter so that the default value can be modified and three overloaded computeBill() methods for a book store:
• When computeBill() receives a single parameter, it represents the price of one book ordered. Add the VAT and return the total due.
• When computeBill() receives two parameters, they represent the price of the book, and the quantity ordered. Multiply the two values, add the VAT and return the total due.
• When computeBill() receives three parameters, they represent the price of the book, the quantity ordered and a voucher value. Multiply the price and quantity, reduce the result by the voucher value, and then add the VAT and return the total due.

For each student, the prompts in Figure 2, Figure 3 and Figure 4 were run for both formative assessments. This was repeated five times to improve consistency and minimise the impact of possible GenAI hallucinations. An example of the result generated by the GenAI grading tool is shown in Figure 5. One at the end of the statement indicates a mark; otherwise, if left blank, no mark was awarded.

Figure 5

GenAI Grading Tool Result

```
{
    " BankAccount (String, int,double)":1,
    " getName() " :1,
    " getID() " :1,
    " getBalance()":1,
    " getClosed() ":1,
    "deposit(double)":1,
    "withdraw(double)":,
    "closeAccount()":,
"transferTo(BankAccount,double)":1

}
```

The total grades for each student's submission were used in the comparison with the marks generated from the automated grading system.

Results and Discussion

To evaluate the statistical differences between the automated grading system and the GenAI grading tool using prompt 1 and prompt 2, a paired sample t-test and a correlation coefficient were used for this study. It is worth noting that prompt 1 does not include the question context, while prompt 2 does include the question context. The total number of student code submissions was 781.

Results

Table 1 presents the individual score for each run, as well as the mean scores for both GenAI prompt 1 and prompt 2 for both formative assessments. The prompts were run separately five times to evaluate the consistency of each prompt. The average score for formative assessment (FA) 1 - prompt 1 across the five runs was 99.39, while FA 1 - prompt 2 achieved an average score of 98.40 across all five runs. The average score for FA 2 - prompt 1 across the five runs was 97.28, while FA 2 - prompt 2 achieved an average score of 92.72 across all five runs.

Table 1

A Comparison of GenAI Prompt 1 and Prompt 2 over Five Runs for Both Formative Assessments

	Formative Assessment 1 (FA 1)		**Formative Assessment 2 (FA 2)**	
Runs	**Prompt 1**	**Prompt 2**	**Prompt 1**	**Prompt 2**
1	99.34	98.43	97.28	93.28
2	99.52	98.66	97.31	91.73
3	99.45	97.72	97.19	92.43
4	99.48	98.82	97.27	93.01
5	99.15	98.37	97.36	93.20
Mean	99.39	98.40	97.28	92.72

Table 2 presents a comparative analysis of the automated grading system against GenAI prompt 1 and prompt 2. The results show the mean of the automated marking and the mean of all the runs using prompt 1 and prompt 2; the difference in means between the automated marking and the means of running prompt 1 and prompt 2; a significance value generated from a T-test comparing both prompts to the automated marking; and a correlation coefficient comparing both prompts to the automated marking.

Table 2

A Comparison of Automated Grading System with GenAI Grading Tool

	Formative Assessment 1			**Formative Assessment 2**		
	Automated Grading System	**GenAI Grading Tool**		**Automated Grading System**	**GenAI Grading Tool**	
Analysis		**Prompt 1**	**Prompt 2**		**Prompt 1**	**Prompt 2**
mean	93.98	99.39	98.40	95.72	97.28	94.79
mean difference	-	5.41	4.42		1.56	3.00
significance	-	8.06E-12	5.94E-09		0.013	1.3415E-05
correl. coefficient	-	0.246	0.326		0.432	0.436

Table 3 shows a comparison similar to that of Table 2 between the automated grading system and GenAI grading tool; however, in this case all grades of zero provided by the automated marker, and their corresponding results from GenAI grading tool, were removed. The reason for this is due to the automated grading system only marking correctly compiled projects. Students whose code was

technically correct but submitted in the incorrect format or with minor syntax errors, would receive zero from the automated grading system, while GenAI grading tool would assess the code.

Table 3

A Comparison of Automated Marking with GenAI, Zeros Removed

Analysis	Formative Assessment 1			Formative Assessment 2		
	Automated Grading System	GenAI Grading Tool		Automated Grading System	GenAI Grading Tool	
		Prompt 1	Prompt 2		Prompt 1	Prompt 2
mean	98.92	98.92	99.60	97.98	99.39	98.61
mean difference	-	-0.678	0.005	-	0.699	3.44
significance	-	0.006	0.984	-	0.377	8.91E-25
correl. coefficient	-	0.361	0.406		0.920	0.616

The distribution of grades from the automated grading system is shown in Table 4.

Table 4

Marks Distribution from the Automated Grading System

Grade	Occurrence
0	39
20	3
40	1
60	5
80	15
100	716

The findings indicate a mixed set of results, showing that, given a set of circumstances, GenAI grading tool performs as well as the automated marking system on efficacy. The weak to moderate, positive correlation for prompt 1 (r=0.361) and prompt 2 (r=0.406) for formative assessment 1, seen in Table 3, indicates that there is a relationship between the automated grading system and the GenAI grading tool. Similarly, for formative assessment 2 (seen in Table 3), prompt 1 (r=0.919) and prompt 2 (r=0.615) show high levels of correlation. These results are discussed next.

Discussion

Several interesting points are worthy of discussion based on the results in the previous section. Firstly, Table 1 shows the consistency of the GenAI grading tool for both prompts. In other words, GenAI hallucinations were low. This could be due to a strong focus on providing GenAI with: (a) clear and specific prompts; (b) enough information; and (c) avoiding ambiguous prompts that could lead to misinterpretation.

Secondly, Table 2, based on the low significance and correlation coefficient between the results of prompt 1 and prompt 2 in comparison to the marks from the automated grading system, shows that the automated grading system and GenAI grading tool results were notably different.

Thirdly, to further identify whether this difference was due to errors in the way that the GenAI grading tool marks the assessments, the individual grades were inspected. It became clear that the automated grading tool marks had a higher bimodality, with most results being either 100% or 0% (see Table 4).

Lastly, to investigate the impact of this bimodality, all grades of zero were removed from the data and the same comparisons were performed. The results of this in Table 3 indicate that the GenAI grading tool performs equally well as the automated marking system (based on means, significance and correlation coefficient). This is due to the fact, as mentioned above, that the automated marking system graded only correctly compiled code.

The difference in results between prompt 1 and prompt 2 for formative assessment 1 (in Table 3) is of interest. The GenAI grading tool with prompt 2 (i.e., with the question context included) more closely matches the automated grading system on all metrics (mean difference, significance, and correlation coefficient). From this, it seems that the inclusion of the question context in the prompt results in grading that is as accurate as that done by the automated grading system. In contrast for formative assessment 2, there is a much larger discrepancy between prompt 2 and the automated grading system. It seems clear that the inclusion of more context changes the result of the GenAI grading tool-based marking process.

Conclusion

The results suggest that GenAI could be a potential alternative to the automated grading system, especially in CS1 modules with large student cohorts. The use of GenAI could save lecturer marking time as well as assist in the development of unit tests and provide potential solutions. It could also enhance the student learning experience by using GenAI as a supportive tool. The findings further underscore

the importance of conducting multiple runs when using GenAI grading tools to ensure consistency and reliability of assessment outcomes, and to mitigate the impact of potential hallucinations. Furthermore, it is interesting to note that providing GenAI with the assessment question, as well as eliminating the zeros, may infer that GenAI grading tools marked as accurately as the automated grading system.

While these insights showcase the advantages GenAI may offer, it is crucial to reflect on broader pedagogical implications. Beyond reducing marking time, educators should also consider how GenAI can enhance fairness and transparency in assessment processes, ensuring that assessment practices are not only efficient but also equitable and explainable, thus providing meaningful feedback to students. Moreover, GenAI has the potential to democratise access to formative assessment while upholding academic standards.

Nonetheless, this study highlights several limitations. The interpretability and transparency of GenAI grading decisions remain an area of concern, especially when complex or subjective criteria form part of the grading process. Further research needs to be conducted to understand how GenAI interprets and evaluates student work when providing more contextual information, including assessment instructions and marking criteria, to name a few. Ethical considerations around fairness, bias, and data privacy should be addressed to ensure responsible adoption. Additionally, the tendency of GenAI to generate inconsistent outputs across multiple runs necessitates further research into strategies that minimise discrepancies and enhance GenAI grading tools robustness.

References

Ala-Mutka, K., Uimonen, T., & Järvinen, H. M. (2004). Supporting students in C++ programming courses with automatic program style assessment. *Journal of Information Technology Education: Research, 3*, 245-262. https://jite.org/documents/Vol3/v3p245-262-135.pdf

Antonucci, P., Estler, C., Nikolić, D., Piccioni, M., & Meyer, B. (2015). An incremental hint system for automated programming assignments. *ITiCSE '15: Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education (*pp. 320 - 325). https://doi.org/10.1145/2729094.2742607

Bang, Y., Cahyawijaya, S., Lee, N., Dai, W., Su, D., Wilie, B., Lovenia, H., Ji, Z., Yu, T., Chung, W., Do, Q. V., Xu, Y., & Fung, P. (2023). A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity. *Computation and Language*. https://doi.org/10.48550/arXiv.2302.04023

Bond, M., Khosravi, H., De Laat, M., Bergdahl, M., Negrea, V., Oxley, E., Pham, P., Chong, S. W., & Siemens, G. (2024). A meta systematic review of artificial intelligence in higher education: A call for increased ethics, collaboration, and rigour. *International Journal of Educational Technology in Higher Education, 21*, 4. https://doi.org/10.1186/s41239-023-00436-z

Chan, H. C. B. (2023). Grading generative AI-based assignments using a 3R framework. In *2023 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE)*, Auckland, (pp. 1-5). DOI**:** 10.1109/TALE56641.2023.10398408

Crompton, H., & Burke, D. (2023). Artificial intelligence in higher education: The state of the field. *International Journal of Educational Technology in Higher Education, 20,* 22. https://doi.org/10.1186/s41239-023-00392-8

Jukiewicz, M. (2024). The future of grading programming assignments in education: The role of ChatGPT in automating the assessment and feedback process. *Thinking Skills and Creativity, 52,* 101522. https://doi.org/10.1016/j.tsc.2024.101522

Kantrowitz, A. (2022, December 2). *Finally, an A.I. chatbot that reliably passes "the Nazi test"*. Slate. https://slate.com/technology/2022/12/chatgpt-openai-artificial-intelligence-chatbot-whoa.html

Mekterovic, I., & Brkic, L. (2017). Setting up automated programming assessment system for higher education database course. *International Journal of Education and Learning Systems, 2, 287-294.*

Rahman, K. A., & Nordin, M. J. (2007). A review on the static analysis approach in the automated programming assessment systems. In *Proceedings of the 2007 National Conference on Programming (pp.* 1–15).

Rahman, M. M., & Watanobe, Y. (2023). ChatGPT for education and research: Opportunities, threats, and strategies. *Applied Science, 13*(9), 5783. https://doi.org/10.3390/app13095783

Raman, A., & Kumar, V. (2022). Programming pedagogy and assessment in the era of AI/ML: A position paper. In V. Choppella & A. Karkere (Eds.), *COMPUTE '22: Proceedings of the 15th Annual ACM India Compute Conference* (pp. 29 - 34). https://doi.org/10.1145/3561833.3561843

Rudolph, J., Tan, S., & Tan S. (2023). ChatGPT: Bullshit spewer or the end of traditional assessments in higher education? *Journal of Applied Learning Teaching, 6*(1), 342-363. https://doi.org/10.37074/jalt.2023.6.1.9

Tobler, S. (2024). Smart grading: A generative AI-based tool for knowledge-grounded answer evaluation in educational assessments. *MethodsX 12, 102531.* https://doi.org/10.1016/j.mex.2023.102531

Ullah, Z., Lajis, A., Jamjoom, M., Altalhi, A., Al-Ghamdi, A., & Saleem, F. (2018). The effect of automatic assessment on novice programming: Strengths and limitations of existing systems. *Computer Applications in Engineering Education, 26*(6), 2328-2341. https://doi.org/10.1002/cae.21974

Van Verth, P. B. (1985). *A system for automatically grading program quality (Metrics, software metrics, program complexity)*. [Doctoral dissertation, Computer Science Department, State University of New York at Buffalo].

Author Details

Pieter Joubert
School of Computer Science, College of Engineering and Physical Sciences
University of Birmingham
p.joubert@bham.ac.uk

Wendy Yanez-Pazmino
School of Computer Science, College of Engineering and Physical Sciences
University of Birmingham, UK
w.yanez@bham.ac.uk

Jacqui Chetty
School of Computer Science, College of Engineering and Physical Sciences
University of Birmingham, UK
j.chetty@bham.ac.uk