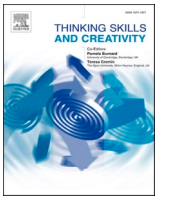
Thinking Skills and Creativity 52 (2024) 101522

Contents lists available at ScienceDirect

Thinking Skills and Creativity

journal homepage: www.elsevier.com/locate/tsc



The future of grading programming assignments in education: The role of ChatGPT in automating the assessment and feedback process

Marcin Jukiewicz

Department of Logic and Cognitive Science, Faculty of Psychology and Cognitive Science, Adam Mickiewicz University, Wieniawskiego 1, Poznan, 61-712, Poland

A R T I C L E I N F O

Keywords:
Improving classroom teaching
21st century abilities
Adult learning
ChatGPT

A B S T R A C T

This research evaluated ChatGPT's potential as a tool for grading programming tasks, exploring its capability to understand and assess code quality. The study took place over a 15-week Python programming course with 67 students of the Cognitive Science program. Nine different assignments were assessed by both a teacher and the ChatGPT system, and the grading differences were recorded. The teacher's grades were higher than those generated by ChatGPT. Despite this, there was a strong positive correlation between these grades, suggesting consistency in grading. Nonetheless, the repeatability of ChatGPT's evaluations was excellent, and the observed differences in successive evaluations during grading iterations were negligible. The study concludes that ChatGPT could be a beneficial tool for grading programming assignments, providing several advantages such as time efficiency, quality assessment, unbiased grading, enforcement of coding standards, and the ability to generate feedback. However, the system has limitations such as cost, potential hallucinations, lack of absolute agreement reproducible results, and the occasional need for teacher intervention. The study suggests that the artificial intelligence model could complement or even substitute human grading but requires careful usage and potential verification by a human teacher.

1. Introduction

Since coding skills have become an essential part of education in the 21st century, the number of people learning programming is rapidly increasing (Nouri et al., 2020). Furthermore, due to the COVID-19 pandemic and the associated changes in education methods, remote learning has become the new norm, accelerating the need for automation in programming assignment grading.

Automating these processes, supported by modern code review systems, peer review techniques, Massive Open Online Courses (MOOC), and e-learning tools, set the direction for developing computer science education (Lo, 2023). In general, three main methods of assessing code written in any programming language can be distinguished in the literature (and practice):

1. Teacher assessment of assignments is a traditional model that relies on the direct knowledge and expertise of the teacher. However, it is a time-consuming process that requires a significant amount of time from the teacher. It is particularly challenging to apply on a large scale in large groups of students or in the case of online courses (Keuning et al., 2019).

E-mail address: marcin.jukiewicz@amu.edu.pl.

https://doi.org/10.1016/j.tsc.2024.101522
Received 31 July 2023; Received in revised form 27 February 2024; Accepted 26 March 2024
Available online 27 March 2024

2. Peer assessment or peer review is another approach that promotes active student participation in the learning process by allowing them to assess the work of others. Despite numerous benefits, such an approach is often prone to subjectivity and unevenness in grading quality (Indriasari et al., 2020; Sun et al., 2019; Wang et al., 2012).
3. Automated unit testing is the third method allowing immediate and objective code evaluation. It is a highly effective tool, although limited to checking specific predefined outcomes rather than the creative process or code quality. This approach is commonly used in MOOC, enabling instant grading and feedback for students, thus increasing the efficiency of the learning process (Douce et al., 2005; González-Calatayud et al., 2021; Skalka et al., 2019).

In 2024, ChatGPT can be identified as a tool for automated programming assignment grading that can bring new perspectives. ChatGPT, created by OpenAI, is a large language model based on the GPT (Generative Pre-trained Transformer) architecture. The model has been trained on diverse textual data from the internet, allowing it to generate coherent and understandable text based on the given information. The model can answer questions, generate long essays, and write creatively, similar to a human (Lund and Wang, 2023).

These models utilize prompts, which are instructions or questions to generate a specific type of response. When formulating effective prompts for models, it is crucial to consider context, precision, and clarity to enable the model to generate the most relevant and helpful answers.

ChatGPT can be used for various purposes, ranging from chat support as a source of information (Biswas, 2023a; 2023b), content generation (Hill-Yardin et al., 2023), to education (Kohnke et al., 2023). However, the potential of this tool is much greater and will be subject to verification in this study by evaluating ChatGPT's capabilities in automatic programming assignment grading.

Automatic code checking brings several advantages, such as:

1. Immediate feedback: Students receive direct feedback on the correctness of their code, enabling them to address issues more quickly and improve their skills (Skalka et al., 2019; Wang et al., 2011).
2. Teacher efficiency: Teachers save time previously spent on checking and correcting repetitive errors, allowing them to focus on more complex teaching aspects (Mekterovic and Brkic, 2017; Skalka et al., 2019).
3. Scalability: Automatic code checking enables teaching and grading large groups of students without increasing the workload for teachers. This is particularly important in the context of large-scale online courses like MOOC (Mekterovic and Brkic, 2017; Skalka et al., 2019).
4. Learning process efficiency: Through error tracking, problem-solving speed, and solution quality, the learning process becomes more efficient. Automatic checking also enables quantification and description of individual parts of the process, aiding in identifying problematic areas (Rahman and Watanobe, 2023; Skalka et al., 2019; Ullah et al., 2018; Wang et al., 2011).
5. Coding style education and testing skills: Some automatic code-checking systems can help teach good coding practices and testing skills (Rahman and Watanobe, 2023; Skalka et al., 2019; Ullah et al., 2018; Wang et al., 2011).
6. Cheating detection: The ability to automatically compare code between different students can assist in detecting and preventing plagiarism (Ullah et al., 2018).
7. Personalized feedback: Automatic code-checking systems can generate personalized comments and suggestions for code improvement, contributing to students' better understanding of programming principles and practices (Rahman and Watanobe, 2023; Skalka et al., 2019).

ChatGPT possesses all of the aforementioned benefits, with its results being different and possibly better than those obtained using other methods. Its advanced natural language processing capabilities allow it to understand code at a level that is not accessible to traditional automatic testing methods (Rahman and Watanobe, 2023). However, it is also essential to be aware of the limitations of language models. ChatGPT can sometimes generate untrue or nonsensical information, a phenomenon known as hallucinations (Bang et al., 2023). The causes can be related to model limitations, the quality of the training data, and specific user queries. Therefore, caution should be exercised when using ChatGPT, as the context and manner of asking questions also influence its responses (Qadir, 2023). This aspect will also be addressed in this article.

Table 1
List of topics discussed during the classes.

Lesson	Topic
If-else instructions	variables, if, elif, else, comparison operators, logical operators, built-in functions
Loops	for, while, break, pass, continue, nested loops, built-in functions,
Random and time modules	random and time modules
Lists, tuples and dictionaries	lists, tuples, dictionaries: creation, indexing, modification, iteration, applications
String module	string module
Functions: basics	functions: definition, invocation/call. parameters, local and global variables
Functions: recursion	recursion
Object-oriented programming	class, object, attributes, methods, encapsulation, polymorphism, inheritance
Graphical User Interface	tkinter: creating the main window, widgets, interface layout, event handling, graphics and pictures

2. Subjects and methods

The study examines the possibilities and effectiveness of using ChatGPT as a tool for programming assignment grading. It seeks to determine whether a large language model (LLM) like ChatGPT can understand and assess the correctness of code and evaluate subtle aspects of the code, such as its quality, readability, efficiency, and adherence to coding standards.

In light of the above, the main research question that is posed and sought to be answered in this study is as follows: "Can ChatGPT effectively assess student work and assign grades that align with the teacher's evaluations?"

The course "Programming," targeting students in the Cognitive Science program, lasted for a period of 15 weeks, with two sessions per week. Each session lasted 1.5 h. Throughout the semester, 67 students participated in the classes. The topics and concepts covered during the sessions are presented in Table 1. The classes were conducted using the Python programming language, version 3.10. Visual Studio Code was used as the programming environment. The course structure and the tools employed were designed to maximize support for the teaching process and cater to the needs of Cognitive Science students.

The research made use of the ChatGPT API, specifically employing the "gpt-3.5-turbo" model (July 20 version). All the analyses were conducted using Python version 3.10, utilizing base libraries including NumPy, SciPy, Pandas, and Matplotlib. The results from nine assignments were analyzed. The number of participants in each assignment and tasks in each assignment varied. Table 2 presents a detailed overview of this data. "Number of students" refers to the count of students who were present during a specific test. "Number of tasks" indicates the quantity of programming tasks each student had to complete. Fluctuations in the student group result from two factors: absences during a particular test, and (in the second half of the semester) dropping out of studies. Therefore, the final tests were completed by a smaller number of students. 25 people completed all 9 projects. 20 people completed 8 projects (one absence), and 16 people completed 7 projects (two absences).

In summary of the presented data, the total number of tasks for assessment was 1579. The teacher evaluated each task once, and the ChatGPT evaluated each task fifteen times. The students' works were assessed fifteen times to check if the AI (Artificial Intelligence) generated the same grades agreement or if they had minor differences. Detailed results will be presented in the further sections of this paper.

The ChatGPT system utilized a specially designed prompt based on the knowledge and experience gathered during the "ChatGPT Prompt Engineering for Developers" course.[1] The modified and used prompt content is presented in the following Listing 1.

The key feature of this prompt is that the AI has to generate its own solution and then compare it with the student's solution. Based on this, the work is evaluated as correct, almost correct, or incorrect and assigned 1, 0.5, or 0 points, respectively (detailed descriptions of these grades are provided in Table 3). This suggestion was introduced in the aforementioned course.

3. Results

To scale the grades consistently, all grades were re-scaled from 0 to 1. Fig. 1 presents a comparison of the mean grades (ChatGPT mean, Teacher mean) and their standard deviations (ChatGPT std, Teacher std) assigned by the teacher and ChatGPT for the nine different assignments. For all assignments, the mean grades assigned by the teacher are slightly higher than the grades generated by ChatGPT. In most tests, the standard deviation is higher for the teacher's grades, indicating a more significant score variation.

The first step was to examine whether the grades assigned by ChatGPT and the teacher were in agreement, meaning that the student's work was assessed similarly in both cases, such as being rated as "correct", "almost correct" or "incorrect". The Pearson correlation test shows a significant relationship ($r \approx 0.81, p < 0.001$) between teacher-assigned grades and ChatGPT-generated grades, indicating a strong positive correlation, with little chance of the observed dependencies being due to chance (Dancey and Reidy, 2007). Therefore, an ANOVA (ANalysis Of VAriance) test with repeated measures was conducted to examine whether there were statistically significant differences in the grades assigned by ChatGPT and the teacher. The test returned a very low corrected p-value (less than 0.001, with a statistic value ≈ 35.44), suggesting that the differences between the teacher's grades and ChatGPT's grades are statistically significant. In the posthoc test with Bonferroni correction, the following significant differences were identified (arranged by the strength of the interaction effect): in task 9 ($p < 0.001$, statistic $= -10.86$), task 8 ($p < 0.001$, statistic $= -6.05$), task 3 ($p < 0.001$, statistic $= -5.82$), task 4 ($p \approx 0.001$, statistic $= -3.43$), task 2 ($p \approx 0.03$, statistic $= -3.11$).

The situation where a high correlation and a low p-value in the ANOVA test may seem somewhat paradoxical, as a high correlation suggests high consistency (for more details about that term, see Stemler, 2019) in the grades, while a low p-value in the ANOVA suggests the existence of significant differences (this is a well-documented phenomenon, see Jönsson and Balan, 2019 and Jönsson et al., 2021 for more details). However, this may be because, despite similar overall grading tendencies (hence the high correlation), there are systematic, significant differences in the detailed assessment between the teacher and the ChatGPT system. For example, the teacher and ChatGPT may generally agree on which students demonstrate a higher level of skills but differ in evaluating specific aspects of the student's work.

Based on these results and the data presented in Fig. 1, it can be concluded that AI, like a human, assesses whether a student's work is "correct", "almost correct" or "incorrect". However, there are differences in the point allocations. ChatGPT assigns fewer points, suggesting it may be perceived as a stricter grader.

In the next step, an attempt was made to examine the specific discrepancies in the grades assigned by ChatGPT and the teacher.

[1] More information about the course is available at https://www.deeplearning.ai/short-courses/chatgpt-prompt-engineering-for-developers/

Table 2
The number of participating students and the topics covered in each assignment.

	Number of students	Number of tasks
If-else instructions	63	1
Loops	66	4
Random module	61	4
Lists, tuples and dictionaries	62	4
String module	59	4
Functions: basics	63	4
Functions: recursion	56	3
Object-oriented programming	53	1
Graphical User Interface	51	1

Below, three different tasks will be presented, differing in terms of teacher grading and ChatGPT evaluation. The first task with the following prompt was given:

> Write a program that displays the 4th row of the multiplication table, which is the sequence of numbers: 4, 8,..., 40, using any loop construct you have learned. Use multiplication and a loop to generate the consecutive elements.

The student's solution is presented in the following Listing 2.

ChatGPT assessed the task with 0 points, while the teacher awarded 1 point (out of 1 possible). ChatGPT provided the following feedback:

> The student's solution is incorrect because the loop condition digit != 11 will result in an infinite loop. The condition should be digit $\leq$ 10 to ensure that the loop iterates ten times. Additionally, the student's solution could be improved using a for loop with the range function, as shown in the solution.

The adjusted student's solution is presented in the following Listing 3

The second task had the following content:

> Your task is to create a simple game that operates as follows:

- The program randomly selects a number from the range 1 to 100.
- The user enters a number of their choice.
- If the player enters a larger number, they receive one point. If the computer's number is larger, the computer receives one point.
- The program then returns to step 2. After ten attempts, the program should display the final game results.

The adjusted student's solution is presented in the following Listing 4

In this case, ChatGPT awarded one point and considered the task solved correctly. The generated feedback is below.

> Your student has provided a correct and functional solution to the problem. The program is structured logically, and it adheres to the requirements of the task. The user is prompted to input their name, and then the game proceeds according to the specified rules. The number of attempts is limited to 10, and the results are displayed at the end.

However, the teacher assessed it as "almost correct" because the code lacks the game results display (After ten attempts, the program should display the final game results). This is an example of ChatGPT not always fully understanding the task description it needs to accomplish.

The content of the third task was as follows:

> Your task is to create a simple game that operates as follows:

- The program randomly selects a number from the range 1 to 6 twice (rolling dice).
- The program "rolls the dice" twice for the user (randomly selects numbers from the range 1 to 6).
- The winner is the one who has rolled the same number twice. If both players or neither player achieves this, no one wins. The winner receives 1 point.
- Then the program returns to step 1. After 10 game repetitions, the program should return the game results.

The adjusted student's solution is presented in the following Listing 5

In this case, ChatGPT assessed the task as "almost correct" and provided feedback:

> The student's solution is somewhat correct. It generates random numbers for dice rolls, checks if the computer or user has rolled the same number twice, and then awards points accordingly. However, there are some improvements that can be made. The student's solution, however, only checks for the same numbers for the computer and user separately. It does not consider the scenario where both players win or neither wins, which is a requirement of the task. The variable names in the student's solution are not descriptive. Consider using more meaningful names for better code readability.

During the assessment of this task, the teacher made a mistake as he evaluated it as completed "correctly". He did not notice what ChatGPT mentions: "It does not consider the scenario where both players win or neither wins, which is a requirement of the task."

```
Your task is to grade the student's solution. To solve the
problem, do the following:
- First, work out your own solution to the problem.
- Then, compare your solution to the student's solution and
evaluate if the student's solution is correct or not.
Don't decide if the student's solution is correct until
you have done the problem yourself. If the task involves
a GUI, separately evaluate the interface and functionality
of the program. Use the following format:
Question:
```
question here
```
Student's solution:
```
student's solution here
```
Actual solution:
```
steps to work out the solution and your solution here
```
Is the student's solution the same as the actual solution
just calculated:
```
yes or no
```
Student grade:
```
correct or almost correct or incorrect
```
Feedback:
```
feedback generated by ChatGPT
```
```

Listing. 1. Comparison of the teacher and ChatGPT's mean scores assigned for each assignment, along with their standard deviations. All scores are presented on a scale from 0 to 1.

Table 3
An extended description of each of the three grades that the student could receive.

Grade	Grade description
Correct	The student fully understood the given task and met all specified requirements. The program runs as expected and fulfills all test conditions.
Almost correct	The student understood some aspects of the task, but certain parts may be unclear or omitted. The code contains shortcomings that affect the program's functionality. The program may work correctly in some cases but fails in others.
Incorrect	The student did not grasp the essence of the task or its key aspects. The code includes serious errors that prevent the program from functioning correctly.

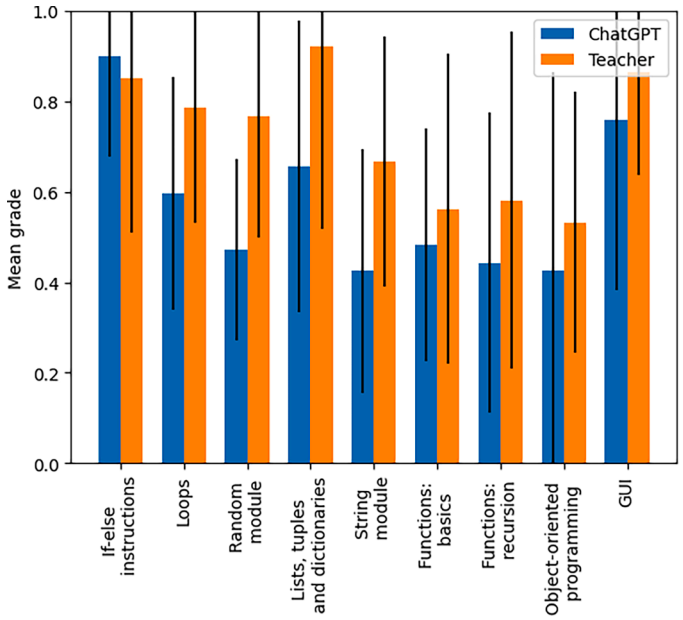


Fig. 1. The content of the used prompt.

```
digit = 1
while digit != 11:
    print(digit * 4)
    digit += 1
```

Listing. 2. The student's solution to the first task.

```
for i in range(1, 11):
    print(i * 4)
```

Listing. 3. The solution to the first task proposed by ChatGPT.

Unfortunately, this is a human error that can occur when evaluating dozens of tasks in a row.

While preparing the above section, the ChatGPT system was repeatedly asked to evaluate the same task. Its responses, mainly the assigned grades, were not always identical. Therefore, it was decided to examine the extent of differences in task grades ChatGPT gave during successive queries. To do this, the Interclass Correlation Coefficient (ICC) was calculated. A value of approximately 0.95 (with significance below 0.001) was obtained, indicating nearly absolute agreement between the repeated evaluations (Koo and Li, 2016). Cohen's d value was also examined to assess the magnitude of the differences between the grades. A value of 0.13 was obtained, suggesting that the differences in successive evaluations are relatively small or negligible (Cohen, 2013), which confirms the results obtained using ICC.

In the final stage, the question was explored whether ChatGPT should assess each task multiple times and whether the final result should be the mode of the grades. A student's assignment evaluation must depend solely on their subject matter proficiency and not on any potential hallucinations of ChatGPT. Therefore, each task was evaluated a total of 15 times. In this case, Cohen's d value was calculated by comparing the grades assigned by the teacher and the mode of the grades given by ChatGPT. The results obtained are presented in Fig. 2. The agreement coefficient initially decreases and then stabilizes. It can be assumed that hallucination ratings will have a negligible impact on the grade only after seven iterations of evaluating a specific assignment.

The discrepancy between the teacher evaluations and the AI observed in this study can be attributed to two factors. The first factor is the "human factor". Teachers often tend to give higher grades for work that are not perfect. This is particularly evident when the code generally functions and meets the task's requirements, although it may not be an optimal solution. The second factor is teacher errors.

```
import random

print("Please enter your name:")
name = input("")

user_point = 0
comp_point = 0
attempts = 0
while attempts < 10:
    print("Please choose a number from the range 1-100:")
    user_number = int(input(""))
    print(f"{name} chose the number: {user_number}")
    random_number = random.randint(1, 100)
    print(f"The computer chose the number: {random_number}")
    if user_number > random_number:
        user_point += 1
    else:
        comp_point += 1
    attempts += 1
```

Listing. 4. The student's solution to the second task.

```
import random
points1 = 0
points2 = 0
for i in range(0, 10):
    dice1 = random.randint(1, 6)
    dice2 = random.randint(1, 6)
    dice3 = random.randint(1, 6)
    dice4 = random.randint(1, 6)
    if dice1 == dice2:
        points1 += 1
    elif dice3 == dice4:
        points2 += 1

if points1 > points2:
    print("Player 1 (computer) won")
elif points2 > points1:
    print("Player 2 (user) won")
else:
    print("No one won")
```

Listing. 5. The student's solution to the third task.

This category includes the teacher's mistakes, usually resulting from inattention or inflated grading.

4. Conclusion

The present study examined the possibility of utilizing ChatGPT for grading programming assessments. The grades assigned by ChatGPT were compared to those given by human teachers. The results indicate a strong positive correlation (agreement) between the

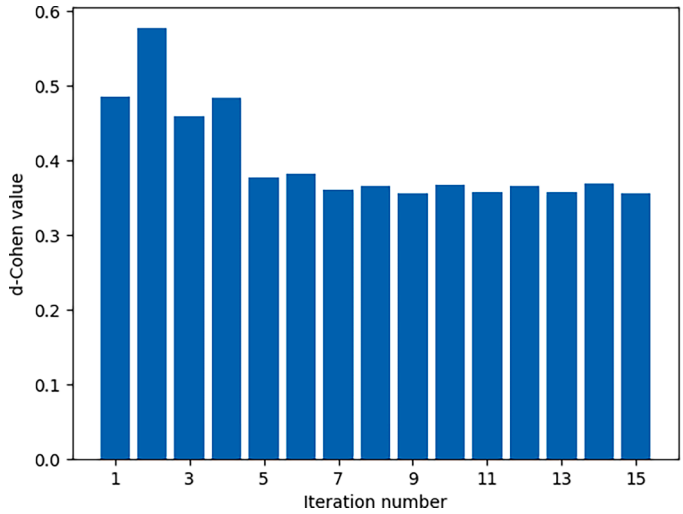


Fig. 2. Comparison of the d-Cohen coefficient values depending on the number of iterations for evaluating a single task.

grades generated by ChatGPT and the human teachers. However, some differences in the assigned points suggest that ChatGPT may be a stricter grader. It is important to note that this comparison is based on a single assessment by a human teacher. These differences could stem from the subjective preferences of the teachers or limitations inherent in the ChatGPT model. In future work, whether teacher evaluations remain constant over time should be specified. This is crucial as it will help determine the objectivity of previously assigned grades. The final answer should ultimately be whether ChatGPT is a more restrictive teacher.

Another important matter worth addressing in future work is an examination of the impact of the prompt on the observed differences. Given their systematic nature, their origin is likely not attributable to the teacher but rather to the calibration of assessments provided by the AI system. This raises the question of whether the calibration process may influence the observed differences.

Based on the conducted analyses, it can be concluded that a language model is a tool that can significantly enhance the work of teachers. The main advantages include:

1. Time efficiency: Teachers require several hours to grade dozens of assessments, while ChatGPT can perform this task within minutes. The average time needed to assess and provide feedback for a single query is approximately 9.5 seconds.
2. Assessment of work quality: Similar to humans, ChatGPT can determine whether a work is correct, mostly correct, or incorrect.
3. Elimination of human bias in grading: ChatGPT is not affected by fatigue and typically does not make grading errors.
4. Emphasis on compliance with coding best practices: The application of AI enforces programming standards from the early stages of education, such as adhering to PEP8 guidelines (van Rossum et al., 2001) in the case of Python, which provide recommendations on formatting and naming conventions to enhance code readability.
5. Objectivity: ChatGPT is impartial and unbiased in its evaluations.
6. Generation of feedback: ChatGPT can provide meaningful feedback for student assignments, all within a few minutes, as mentioned in the previous point. Teachers would require several additional hours to accomplish the same task.

The last point is relevant for comparing AI and human teachers and AI with other automated code assessment tools, as none can provide such comprehensive feedback. Therefore, ChatGPT seems to be the most accurate tool for code assessment and feedback generation. However, there are some limitations to consider:

1. Cost: The ChatGPT API is a paid service. The average cost of checking one assignment is 0.22 USD (July 2023).
2. ChatGPT hallucinations: Sometimes, the model identifies errors in a student's solution where none exist in reality.
3. Reproducible results: ChatGPT does not always return identical grades. Although the statistical differences in grading between the two queries are small, it is crucial for teachers to ensure a reliable evaluation of the student's work and to avoid AI hallucinations during grading. As the results demonstrate, this issue can be minimized by submitting queries to the model at least seven or more times.
4. Requirement for teacher intervention: In the event of a grading error by ChatGPT that disadvantages the student, teacher intervention is necessary.

In conclusion, ChatGPT can potentially be a helpful tool for the automated grading of programming assignments, but it requires careful and conscious usage. It can be seen as a complement to human teacher evaluations or even as a substitute. In the latter case, the grades should be verified by a teacher with more comprehensive programming knowledge and experience. Otherwise, ChatGPT could enhance existing code assessment and grading tools, specifically in generating feedback.

CRediT authorship contribution statement

Marcin Jukiewicz: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Declaration of Competing Interest

None.

Data availability

Data will be made available on request.

Acknowledgment

The author would like to thank Pawel Kleka, Pawel Lupkowski, Dawid Ratajczyk and Michal Wyrwa for helpful feedback and comments on a draft of this paper.

References

Bang, Y., Cahyawijaya, S., Lee, N., Dai, W., Su, D., Wilie, B., Lovenia, H., Ji, Z., Yu, T., Chung, W. et al. (2023). A multitask, multilingual, multimodal evaluation of Chat GPT on reasoning, hallucination, and interactivity. arXiv preprint arXiv:2302.04023.

Biswas, S. S. (2023a). Potential use of chat gpt in global warming. *Annals of Biomedical Engineering, 51*(6), 1126–1127.

Biswas, S. S. (2023b). Role of chat gpt in public health. *Annals of Biomedical Engineering, 51*(5), 868–869.

Cohen, J. (2013). *Statistical power analysis for the behavioral sciences*. Cambridge, Massachusetts: Academic press.

Dancey, C. P., & Reidy, J. (2007). *Statistics without maths for psychology*. New York: Pearson education.

Douce, C., Livingstone, D., & Orwell, J. (2005). Automatic test-based assessment of programming: A review. *Journal on Educational Resources in Computing (JERIC), 5*(3), 4–es.

González-Calatayud, V., Prendes-Espinosa, P., & Roig-Vila, R. (2021). Artificial intelligence for student assessment: A systematic review. *Applied Sciences, 11*(12), 5467.

Hill-Yardin, E. L., Hutchinson, M. R., Laycock, R., & Spencer, S. J. (2023). A chat (GPT) about the future of scientific publishing. *Brain, Behavior, and Immunity, 110*, 152–154.

Indriasari, T. D., Luxton-Reilly, A., & Denny, P. (2020). A review of peer code review in higher education. *ACM Transactions on Computing Education (TOCE), 20*(3), 1–25.

Jönsson, A., & Balan, A. (2019). Analytic or holistic: A study of agreement between different grading models. *Practical Assessment, Research, and Evaluation, 23*(1), 12.

Jönsson, A., Balan, A., & Hartell, E. (2021). Analytic or holistic? A study about how to increase the agreement in teachers grading. *Assessment in Education: Principles, Policy & Practice, 28*(3), 212–227.

Keuning, H., Heeren, B., & Jeuring, J. (2019). How teachers would help students to improve their code. *Proceedings of the 2019 ACM conference on innovation and technology in computer science education* (pp. 119–125).

Kohnke, L., Moorhouse, B. L., & Zou, D. (2023). ChatGPT for language teaching and learning. *RELC Journal*.00336882231162868

Koo, T. K., & Li, M. Y. (2016). A guideline of selecting and reporting intraclass correlation coefficients for reliability research. *Journal of Chiropractic Medicine, 15*(2), 155–163.

Lo, C. K. (2023). What is the impact of chatGPT on education? A rapid review of the literature. *Education Sciences, 13*(4), 410.

Lund, B. D., & Wang, T. (2023). Chatting about chatGPT: how may AI and GPT impact academia and libraries? *Library Hi Tech News, 40*(3), 26–29.

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

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in k-9. *Education Inquiry, 11*(1), 1–17.

Qadir, J. (2023). Engineering education in the era of chatGPT: Promise and pitfalls of generative AI for education. *2023 IEEE Global engineering education conference (EDUCON)* (pp. 1–9). IEEE.

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

Skalka, J., Drlík, M., & Obonya, J. (2019). Automated assessment in learning and teaching programming languages using virtual learning environment. *2019 IEEE Global engineering education conference (EDUCON)* (pp. 689–697). IEEE.

Stemler, S. E. (2019). A comparison of consensus, consistency, and measurement approaches to estimating interrater reliability. *Practical Assessment, Research, and Evaluation, 9*(1), 4.

Sun, Q., Wu, J., Rong, W., & Liu, W. (2019). Formative assessment of programming language learning based on peer code review: Implementation and experience report. *Tsinghua Science and Technology, 24*(4), 423–434.

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.

van Rossum, G., Warsaw, B., & Coghlan, N. (2001). Style Guide for Python Code. *PEP 8*.https://www.python.org/dev/peps/pep-0008/

Wang, T., Su, X., Ma, P., Wang, Y., & Wang, K. (2011). Ability-training-oriented automated assessment in introductory programming course. *Computers & Education, 56*(1), 220–226.

Wang, Y., Li, H., Feng, Y., Jiang, Y., & Liu, Y. (2012). Assessment of programming language learning based on peer code review model: Implementation and experience report. *Computers & Education, 59*(2), 412–422.