Evaluating ChatGPT for Grading Programming Assignments: Effectiveness, Fairness, and Student Perceptions

Abedallah Zaid Abualkishik[1], Sherzod Turaev[2], Ali A. Alwan[3], Mohamed Elhoseny[4], Mohsin Murtaza[5]
American University in the Emirates, Dubai, United Arab Emirates[1]
United Arab Emirates University, Al Ain, United Arab Emirates[2]
School of Theoretical & Applied Science, Ramapo College of New Jersey, Mahwah, NJ 07430, USA[3]
College of Computing and Informatics, University of Sharjah, 27272, Sharjah, United Arab Emirates[4]
College of Business, Al Ain University, Abu Dhabi Campus, United Arab Emirates[5]

***Abstract*—This study investigates ChatGPT as an automated grading tool for programming assignments in higher education. Three datasets comprising Python, C++, and Java assignments were graded three times by ChatGPT and compared with faculty evaluations. Results show that ChatGPT achieves high grading accuracy, closely aligning with faculty scores and demonstrating statistically significant correlations. Statistical analyses using the Kolmogorov–Smirnov test, paired t-test, and Wilcoxon signed-rank test confirm overall agreement, although ChatGPT tends to apply stricter grading criteria. High intraclass correlation coefficients further indicate strong reliability and consistency across repeated grading attempts. The study highlights the critical role of well-defined rubrics in improving grading alignment and proposes an Instructor–AI Collaborative Rubric Development framework to support effective AI integration in assessment. A survey of 158 students indicates increased satisfaction and trust following disclosure of AI-assisted grading, although some still prefer human evaluation. Overall, the findings provide strong evidence that ChatGPT is a reliable and consistent grading tool, demonstrating close alignment with faculty evaluations and high reproducibility across attempts. However, its effectiveness is critically dependent on well-defined rubrics and requires human oversight to mitigate strictness, ensure fairness, and account for contextual nuances. These results strongly support a hybrid AI–human grading approach, grounded in transparent rubric design and reinforced by appropriate ethical safeguards.**

Keywords*—AI-assisted grading; ChatGPT; automated grading; programming assignments; higher education; grading reliability; rubric-based evaluation

I. Introduction

Automated grading began in the 1970s [1] when educational institutions started using optical mark recognition (OMR) systems for standardized tests, mainly to score multiple-choice questions. These systems became widely used due to their ability to quickly process large volumes of standardized tests. In the 1990s, Intelligent Tutoring Systems (ITS) emerged [2], capable of providing limited feedback on short answers and specific types of student inputs. Over time, ITS evolved to include more sophisticated models, including domain knowledge, student modelling, and pedagogical strategies. ITS systems like LISP Tutor [3], which helped students learn programming languages, played a key role in this development.

Since the 2000s, the rise of machine learning and Natural Language Processing (NLP) has led to more sophisticated systems for assessing essays [4], [5], [6], [7] and programming assignments [8], [9], [10]. Early AI-driven grading tools used NLP to evaluate written work, though they initially faced challenges in accuracy and nuance. Over time, models like ETS's e-Rater and later Large Language Models (LLMs) like ChatGPT became more effective in understanding context, language flow, and even code evaluation.

The rise of LLMs, such as ChatGPT, has brought unprecedented change across many fields, including education and learning. LLMs are a type of artificial intelligence that utilizes deep learning techniques to understand and generate human-like text. They are built upon neural network architectures, primarily transformer models, which allow them to process vast amounts of text data efficiently.

The development of models like OpenAI's ChatGPT marks a significant milestone in NLP, demonstrating capabilities in generating coherent text, understanding context, and engaging in conversations. These models can be leveraged for various educational applications, including automated grading assessment and feedback generation, personalized tutoring, content creation summaries, linguistics, language learning and translation, question generation and answering, making them valuable tools in both traditional and online learning environments [11].

Grading and assessing students' submissions is crucial for both learners and instructors, as it supports scalability in class size, ensures consistency and objectivity in grading, provides immediate feedback, reduces administrative workload for instructors, and enhances the overall learning experience. LLMs like ChatGPT have demonstrated strong capabilities in automating the grading of various types of assignments across multiple domains.

This research aims to evaluate the potential of LLMs, particularly ChatGPT, in assessing programming assignments in Python, C++, and Java programming languages, as follows:

Compare the correctness of grades assigned by ChatGPT versus faculty.

Evaluate the consistency of ChatGPT across several assignments.

Assess the adherence of ChatGPT according to the grading instructions and rubrics.

Gather qualitative feedback from students about ChatGPT grading.

This study builds upon prior research on ChatGPT-based grading by evaluating its performance across three programming languages: Python, C++, and Java, using datasets collected from three different academic institutions. It introduces an Instructor–AI collaborative rubric development framework to support alignment between grading outputs and course learning outcomes. A range of statistical methods, including Kolmogorov-Smirnov, Paired t-test, Wilcoxon Signed-Rank test, and Intraclass Correlation Coefficient (ICC), are applied to assess the accuracy and consistency of ChatGPT's grading. The study also includes a pre/post-reveal student perception survey with open-ended responses to explore students' attitudes toward AI-generated feedback.

The rest of this study is organized as follows: Section II presents the literature review, Section III describes the methods, Section IV presents the results and discussion, Section V discusses the considerations and recommendations, and limitations. Finally, Section VI presents the conclusion of the study.

II. Related Work

Matthews et al. [12] developed and implemented an adaptive learning and grading system aimed at improving the quality, quantity, and speed of feedback. The tool was built using cognitive, behavioral, and grading rubrics for computer literacy assignments. The study tested three hypotheses and found that the system increased feedback quantity and reduced grading time but had no significant effect on feedback quality, highlighting areas for future research. Geigle et al. [13] conducted a systematic study on automating grading for complex assignments, using medical case assessments as a test case. The authors proposed a supervised learning approach with three feature representations to automate grading. Their experiments demonstrated the feasibility of automating grading with examples provided by the instructor. Additionally, they developed a pairwise online active learning strategy, which demonstrated effectiveness in minimizing human grading effort while maintaining grading accuracy.

Wang et al. [14] presented a novel automatic essay scoring system that utilizes NLP and Deep Learning. The system encodes essays as sequential embeddings and employs a bi-directional Long Short-Term Memory (LSTM) to capture semantic information. An attention mechanism was incorporated to help the system focus on critical content, enhancing prediction accuracy. Tested on the Automated Student Assessment Prize dataset, the system achieved state-of-the-art performance by analyzing key sentences and semantic relationships, offering interpretable grades based on essay content. Burrows et al. [15] provided a comprehensive review of Automatic Short Answer Grading (ASAG) systems, noting a significant increase in research, accumulating over 80 papers. The authors identified 35 ASAG systems and categorized them into five temporal themes reflecting methodological and evaluative advancements. They also analyzed six common dimensions of ASAG systems, ranging from preprocessing to effectiveness. A key conclusion indicated that the field entered an "era of evaluation", essential for consolidating ASAG research efforts. Zhang et al. [16] developed an automatic short-answer grading model, addressing challenges in grading semi-open-ended questions common in reading comprehension assessments. Their model integrated both domain-general and domain-specific information, employing LSTM recurrent neural network to capture word sequence information. Experimental results on seven reading comprehension questions with over 16,000 short-answer samples demonstrated that their proposed model outperformed existing grading methods, showcasing its effectiveness in providing accurate assessments.

Grading complex programming assignments at the graduate level is challenging and time-consuming, often requiring up to an hour per student. Novak and Kermek [17] explored automation strategies, such as unit testing and scripting, to improve efficiency without sacrificing quality. They outlined an assessment process that includes preparation, similarity detection, correctness checks, and grading. However, they also highlighted that complete automation is not feasible due to the nuanced nature of programming tasks. Suleiman et al. [18] presented an automated testing tool that enhances the assessment of web application development by reducing manual grading time. Unlike traditional tools, it utilizes modern frameworks, such as Vue.js, Node.js, and MongoDB, to automate code marking, generate test cases, and provide feedback. This tool improves grading efficiency and accurately distinguishes between student submissions and standard answers, addressing the limitations of existing line-by-line comparison methods in educational technology.

Jordan et al. [19] presented a tool that utilizes LLMs to automate personalized feedback based on instructor-defined criteria for open-ended questions. The tool enabled rapid feedback for students to identify knowledge gaps while maintaining depth. The tool is available as an open-source web application and Jupyter Notebook widget, showcasing LLMs' potential to enhance learning outcomes and instructional practices when guided by educators.

Montella et al. [20] presented GAMAI, an AI-powered tool that enhances gamified programming education within the Framework for Gamified Programming Education (FGPE). By automating the generation of gamified exercise scenarios using storytelling and large language models, GAMAI significantly reduces the effort required from instructors. Evaluation results indicated that most AI-generated exercises were ready for use with minimal human input and received positive feedback from students. This research promotes a more efficient and engaging approach to programming education by integrating advanced language models with gamification principles. Messer et al. [21] analyzed 121 papers on automated grading and feedback tools for programming education from 2017 to 2021. The review categorized tools based on assessed skills, approaches, and degrees of automation, finding a predominant focus on object-oriented languages. Most tools employed dynamic techniques, particularly unit testing, for grading, offering limited feedback regarding test outcomes and comparisons to reference solutions.

While fully automated assessments provided rapid feedback and multiple resubmission opportunities, few tools evaluated code maintainability or readability. Evaluation methods primarily relied on student surveys and comparisons with human grading, though the lack of accessible datasets hindered reproducibility.

Jukiewicz [22] evaluated ChatGPT's effectiveness in grading programming assignments during a 15-week Python course with 67 Cognitive Science students. Nine assignments were graded by both a teacher and ChatGPT, revealing that while the teacher's grades were higher, a strong positive correlation indicated grading consistency. ChatGPT demonstrated excellent repeatability, with negligible differences in evaluations. The findings suggest that ChatGPT can enhance grading efficiency, provide unbiased assessments, and generate feedback, though limitations exist, such as costs, occasional inaccuracies, and the need for teacher oversight.

Bengtsson and Kaliff [23] investigated how task context impacts the accuracy of OpenAI's GPT-4 model in grading programming assignments from an introductory programming course. Errors were intentionally injected into assignments, categorized into logical errors like looping and recursion. Results indicated that while context can improve feedback accuracy, it often leads to fewer identified errors, reducing overall assessment accuracy; however, accurate feedback was generated. The study recommends further exploration of context types and the effects of grading repetition on model performance due to its non-deterministic nature.

Iria et al. [24] evaluated ChatGPT and Bard's effectiveness in providing feedback on student exercises in a university programming course focused on concurrency. The study found that neither LLM accurately identified common concurrency errors, achieving only a 50% accuracy rate compared to expert evaluations. Despite the enthusiasm for LLMs in education, their limitations in assessing complex programming tasks highlight the need for caution in their application for specific educational assessments.

Existing research on automated grading and AI-assisted assessment has shown notable progress but still faces key shortcomings. Many studies are limited to specific languages, datasets, or assignment types, which restricts the generalizability of their results. Others provide descriptive evaluations without critically examining issues such as rubric dependence, grading stability, or bias. In several cases, model performance has also proven sensitive to prompt structure and contextual variation, affecting grading consistency and fairness.

This study addresses these limitations by employing multiple datasets across different programming languages, applying repeated grading to evaluate consistency, and using a standardized rubric framework to ensure fairness, transparency, and reproducibility in AI-assisted assessment.

III. Methodology

To address the research questions, three datasets were constructed from Introduction to Programming and Object-Oriented Programming (OOP) courses across three different academic institutions. These datasets comprised assessments written in three programming languages: 146 Python assignments, 188 C++ assignments, and 200 Java assignments. The evaluation of ChatGPT's grading accuracy was assessed by replicating the grading process used by faculty. A detailed rubric, aligned with the course level expectations and learning outcomes, was developed by the instructor to assess the completeness and correctness of students' solutions. This same rubric was then provided to ChatGPT to evaluate its adherence to grading instructions to ensure a direct and fair comparison between human and AI-generated assessments.

All grading was performed using the GPT-4o model via the OpenAI API. To ensure deterministic and reproducible outputs, the temperature was set to 0.0, top-p to 1.0, and other generation parameters were kept at their default settings. The assessment questions ranged from intermediate to challenging. The topics included the basic structure of programming languages and the main concepts of OOP. The topics are outlined in Table I.

The rubrics were used by ChatGPT and faculty to evaluate the correctness, completeness and grade deduction as recommended in [25]. A rubric is a scoring guide used to assess the quality of a student's response or performance. It outlines specific criteria and standards for evaluating different aspects of the work, providing a structured, consistent and replicable method of grading. Table II presents a rubric for grading the question below, taken from [26]. The question is worth 4 marks:

The Fibonacci numbers are defined by the sequence Fibonacci numbers describe the growth of a rabbit population.

f1 = 1

f2 = 1

fn = f(n–1) + f(n–2)

Reformulate that as

fold1 = 1

fold2 = 1

fnew = fold1 + fold2

After that, discard *fold2*, which is no longer needed, and set *fold2* to *fold1* and *fold1* to *fnew*. Repeat an appropriate number of times. Implement a program that prompts the user for an integer *n* and prints the nth Fibonacci number, using the above algorithm. The samples were graded by faculty once, and the same assignment was graded three times by GPT-4o to ensure consistency. The average of the three attempts was used to compare the accuracy of GPT-4o against faculty grades. To achieve the objectives of this study, the following research methods were employed, as shown in Table III.

A. *Prompt Engineering*

ChatGPT’s grading behavior is highly sensitive to prompt wording. To ensure consistency, objectivity, and alignment with the grading rubric, the following strategies were implemented in every grading attempt:

1) Structured Prompt Template: A standardized and pre-defined prompt was used across all grading sessions to minimize variability. The prompt explicitly included:

- A complete rubric with clearly defined criteria (e.g., correctness, efficiency, modularity, error handling).
- The specific weight assigned to each criterion.

- Descriptors for each performance level (e.g., "Exceeds expectations", "Meets expectations").

Example Prompt Used: You are acting as a programming instructor assistant. Your task is to objectively evaluate the following student's solution to a programming assignment using the provided grading rubric. Assess each criterion independently, assign a numerical score, and provide clear, constructive feedback aligned with the rubric descriptors.

- Grading Rubric

Criterion 1: Completeness (Weight: 50%)

Assess whether the solution fully implements the required functionality, addresses all specified requirements, and includes necessary components (e.g., input/output handling, logic structure).

Criterion 2: Correctness (Weight: 50%)

Evaluate whether the solution produces the correct output for a variety of test cases, including edge cases, and whether the implementation logic is accurate and error-free.

TABLE I. LIST OF ASSESSED TOPICS

Topic	Subtopic
Programming with numbers	Variables and arithmetic
Decision	If statement, relational operator, nested branches, Boolean variables, strings, input and output
Repetition	While, for, nested loops, sentinel value, Random numbers
Functions	Function parameters, return, scope, recursion
Lists	Built-in methods, slicing, list algorithms
Files and exception handling	Reading and writing from various files, command line arguments, Exception handling.
OOP	Class, objects, methods and attributes, encapsulation, inheritance, super and sub-classes, overriding, polymorphism.

TABLE II. GRADING RUBRIC FOR CHATGPT AND FACULTY

Criteria	Above Expectations (90%-100%)	Meets Expectations (70%-89%)	Approach Expectations (60% – 69%)	Below Expectations (0 – 59%)
Completeness	Accurately implements the program to prompt for n, computes the nth Fibonacci number using *fold1*, *fold2*, and *fnew*, and updates variables as described. The program handles edge cases (e.g., n=1, n=2) and validates input.	Program is mostly complete, implements the Fibonacci logic with minor omissions or issues, such as missing edge case handling, but works for general inputs.	Program makes significant attempts but fails to fully implement the logic or handle input correctly. May be incomplete or lack certain steps (e.g., variable updates).	Program is incomplete or lacks a coherent approach to calculating the nth Fibonacci number.
Correctness	Produces correct Fibonacci numbers for any valid input n, with no logical or syntax errors. Updates fold1, fold2, and fnew appropriately, and handles iteration correctly.	Generally, correct, but may have minor errors (e.g., off-by-one error) or incorrect variable handling, though the program runs for most inputs.	Several logic or syntax errors present, leading to incorrect Fibonacci outputs or incorrect iteration, but the attempt is partially functional.	Produces incorrect results or fails to run due to major logic or syntax issues. The iterative structure is incorrect, or variables are not updated properly.

TABLE III. RESEARCH QUESTIONS ADDRESSING THE METHOD

Objectives	Method	Data Samples	Statistical Test	Analysis
Compare the correctness of grades assigned by ChatGPT versus faculty.	Compare grades assigned by faculty and ChatGPT using rubrics.	146 Python programming assignments. 188 C++ programming assignments. 200 Java programming assignments.	Descriptive statistics Two Sample Kolmogorov-Smirnov (K-S) test Paired t-test	Accuracy analysis to determine how closely ChatGPT's grades match those given by the faculty using the rubric.
Evaluate the consistency of ChatGPT grading.	Multiple grading attempts of the same assignments by ChatGPT.	146 programming assignments 188 C++ programming assignments. 200 Java programming assignments.	Intraclass correlation coefficient (ICC). Descriptive statistics Two Sample Kolmogorov-Smirnov (K-S) test	Consistency analysis to check if ChatGPT provides repeatable and reliable grading outcomes across different attempts.
Assess ChatGPT's adherence to grading instructions and rubrics.	Four different rubrics were used to evaluate the same solution.	One sample	NA	Comparison of adherence between different rubric-based grading by ChatGPT.
Gather qualitative feedback from students about ChatGPT grading.	Two surveys pre-reveal and post-reveal grading using ChatGPT.	158 surveyed students.	Descriptive statistics Wilcoxon Signed-Rank test	Qualitative feedback analysis to assess student satisfaction. Compare students' responses.

Provide: A numerical score (0–100) reflecting the total grade based on the weighted criteria.

A breakdown of the score for each criterion. Detailed feedback for each criterion, including strengths and areas for improvement.

2) Manual prompt calibration: Before conducting this study, the prompt had been tested on a small pilot set. Responses were compared to faculty expectations to assess clarity and alignment with rubric interpretation. Based on this review, the prompt was refined to reduce ambiguity and improve precision.

3) LLM version control: All grading was performed using the same version of ChatGPT (GPT4o). This helped ensure that no version drift or randomness affected the grading outcomes.

4) Prompt persistence and session context: Each assignment was graded within a consistent chat thread, rather than starting a new session for each attempt. This strategy leveraged ChatGPT's short-term context memory to maintain a consistent interpretation of instructions, contributing to reproducibility across attempts.

IV. Results and Discussion

The empirical analysis was conducted using 146 Python, 188 C++, and 200 Java programming assignments, with all grades normalized to a 100% scale for consistency. Below is the breakdown for the research questions:

A. Research Question 1

1) RQ1: Compare the correctness of grades assigned by ChatGPT versus faculty: Descriptive statistics were employed to summarize the data, while one-sample and Two-Sample Kolmogorov-Smirnov (k-S) tests were conducted to assess normality and distribution. Additionally, a Paired t-test was utilized to evaluate differences in mean scores between faculty and ChatGPT grading. Table IV shows the descriptive statistics of the faculty score, GPT1 score, GPT2 score, GPT3 score, and the average of GPT's scores.

For the Python dataset, the mean scores across the three GPT-generated attempts were consistent, with GPT1 having a slightly higher mean (76.5%) compared to GPT2 and GPT3 (both at 75.1%). The averaged GPT's score was 75.6%, which closely aligns with GPT1, GPT2, and GPT3 scores, reflecting minimal variation across attempts. The results suggest that the GPT grading was consistent between the second and third grading attempts, with a slight change from the first grading attempt. The faculty scores have a slightly higher mean of 77.6%, which is around 2 points higher than the GPT average. This suggests that, on average, ChatGPT was stricter than the faculty in grading. The standard deviation (STD) of the scores was quite similar for both the GPT scores and faculty scores, ranging from 17.03 to 17.35. This indicates that both the GPT and faculty grades had similar levels of variability. The consistency in variability suggests that GPT grading mimics the level of diversity seen in faculty grading following the same grading rubric.

TABLE IV. Descriptive Statistics for Experiment Variables

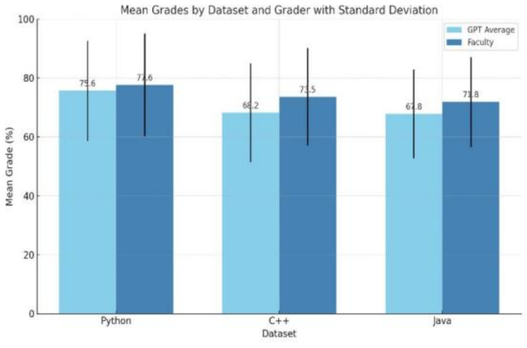
	Mean	SD	MIN	25%	50%	75%	MAX
Python dataset							
GPT1	76.5	17.17	24	65	80	88	100
GPT2	75.1	17.10	20	65	76	87	100
GPT3	75.1	17.10	20	66	76	87	100
GPT average	75.6	17.03	21	65	77	88	100
Faculty	77.6	17.35	16	67	80	92	100
C++ dataset							
GPT1	65.6	17.4	5	55	68	77	100
GPT2	68.6	16.8	5	61	71	80	100
GPT3	70.4	16.4	10	63	74	80	100
GPT average	68.2	16.7	6	60	70	79	100
Faculty	73.5	16.6	12	65	75	85	100
Java dataset							
GPT1	65.9	15.3	15	58	67	75.2	100
GPT2	68.4	15.1	16	59.7	70	78	100
GPT3	69.3	15.3	16	60	71	80	100
GPT average	67.8	15.1	15	59	69	77	100
Faculty	71.8	15.3	19	61	72	82	100

In the Python dataset, the close alignment of the descriptive statistics between GPT scores and faculty scores suggests that ChatGPT can provide grades that are relatively comparable to human grading in terms of central tendency and spread. The minimal difference between the three GPT attempts shows that ChatGPT produces stable grading results across multiple grading instances. This consistency could be valuable when using GPT as a tool to assist in grading.

For the C++ dataset, the results reveal that GPT scores are lower than faculty grades by about 5.3 points on average. The median and quartile values (e.g., Q3: 79 vs. 85) also show a consistent under-grading pattern. Furthermore, the minimum GPT grade is extremely low (6.6), suggesting a few assignments may have been misunderstood or unfairly penalized by GPT. The results reflect an alignment with faculty scores. For the Java dataset, again, GPT under-graded students slightly (~4 points lower on average), the score distribution is similar, as reflected in nearly identical standard deviations. The gap between GPT and faculty is more uniform, suggesting slightly conservative grading but with good consistency.

The analysis of the three datasets across Python, C++, and Java indicates that ChatGPT demonstrates grading behavior that closely approximates faculty-assigned grades in terms of correctness, with consistent patterns observed across all datasets, as shown in Fig. 1. While the average scores assigned by ChatGPT are slightly lower than those given by faculty, typically by 1 to 4 points. This difference remains within an acceptable margin of variability for subjective assessments like programming tasks.

The similarity in standard deviations and medians further supports the reliability of ChatGPT's grading, suggesting it captures student performance with a level of consistency comparable to human instructors. The consistently high maximum scores (100) in both GPT and faculty data suggest that ChatGPT is capable of identifying excellent submissions accurately.



Error bars represent standard deviation. GPT Average represents the mean of GPT-1, GPT-2, and GPT-3 outputs.

Fig. 1. Mean grades for the examined datasets.

However, ChatGPT appears to be slightly more conservative in scoring, particularly on lower-performing assignments, possibly due to its systematic evaluation criteria. The more uniform behavior of GPT models across datasets might reflect systematic grading logic, which contrasts with individual faculty biases or preferences. Overall, these findings suggest that ChatGPT can serve as a reliable dependable grading assistant. Based on the descriptive statistics results, ChatGPT can be considered a credible grading assistant, particularly for Python and Java programming courses. The Pearson Correlation was used to show how linearly the two scores are related. A high correlation closer to 1 would indicate that ChatGPT's grading is consistent with faculty grading, a lower correlation would suggest an absence of a linear relation between both. Fig. 2 shows the Pearson correlation of ChatGPT's first, second, third, and the averaged grade against the faculty grade.

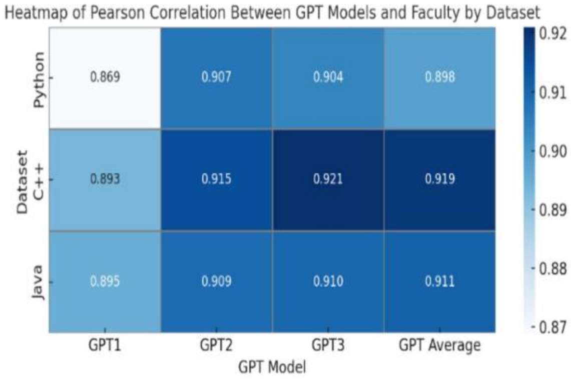


Fig. 2. Pearson correlation of ChatGPT against faculty.

All the GPT attempts show a strong correlation against faculty scores in the three tested datasets. The p-value for all pairs is less than 0.0001, which indicates that the observed correlations are statistically significant and not due to random chance. In other words, we can confidently say that the relationship between GPT grades and faculty grades is real and not a result of randomness. All values are above 0.86, indicating a strong positive linear relationship between ChatGPT-assigned grades and faculty grades. This suggests that ChatGPT grading is highly aligned with human grading, especially in terms of ranking student performance. In all datasets, the second and third grading attempts show higher correlation than the first one. This supports the idea that repeating the grading process with ChatGPT can improve alignment and reliability. The results also suggest that multiple attempts improve correlation, indicating that ChatGPT is more consistent and reliable when used repeatedly. Fig. 3 shows the Bland-Altman plot comparing faculty scores with the average of three GPT grading attempts for the Python dataset. While the Pearson correlation was high (r = 0.898), indicating strong alignment, the plot reveals a slight positive bias (+0.37) and limits of agreement ranging from −9.27 to +10.01. This suggests that GPT grading generally follows faculty trends but may differ by up to ±10 points in individual cases.

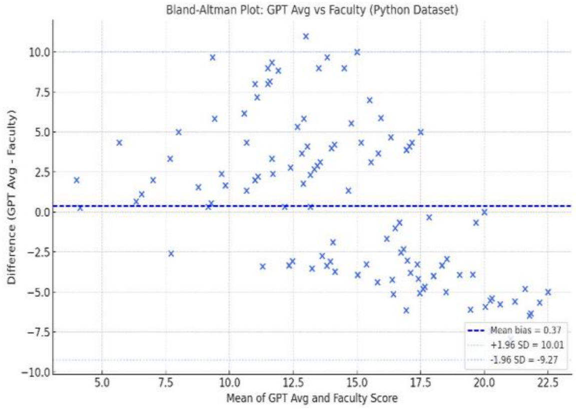


Fig. 3. Bland-Altman plot of GPT average vs. faculty scores (Python dataset).

Fig. 4 shows the Bland-Altman plot comparing faculty scores with the average of three GPT grading attempts for the C++ dataset. Despite a strong Pearson correlation (r = 0.919), the plot reveals a negative bias of −5.30, indicating that GPT tended to score lower than faculty. The limits of agreement (−18.45 to +7.84) suggest notable variability in individual score differences, highlighting that high correlation does not necessarily imply close agreement.

Fig. 5 shows the Bland-Altman plot comparing faculty scores with the average of three GPT grading attempts for the Java dataset. Although the Pearson correlation was strong (r = 0.910), the plot reveals a negative bias of −4.00, indicating GPT systematically assigned lower scores. The limits of agreement (−16.62 to +8.61) indicate moderate variability, suggesting that while GPT followed faculty scoring trends, individual scores often diverged.

The one-sample Kolmogorov-Smirnov test was used to test whether faculty score and GPT's scores are normally distributed. The p-values for GPT1, GPT2, GPT3, GPT average, and Faculty are: 0.092, 0.095, 0.097, 0.095, 0.1. With a p-value greater than 0.05, it suggests that there is insufficient evidence to conclude that the data significantly deviates from the expected normal distribution, as shown in Table V.

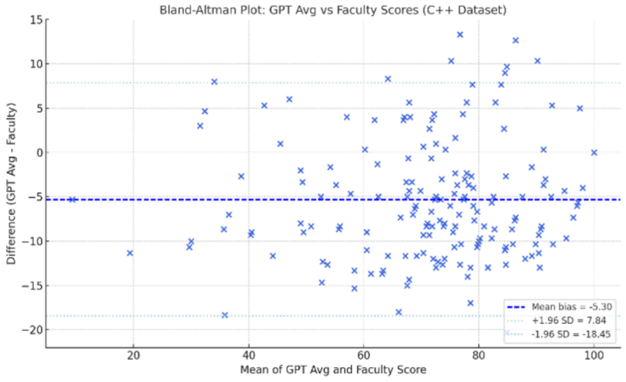


Fig. 4. Bland-Altman plot of GPT average vs. faculty scores (C++ dataset).

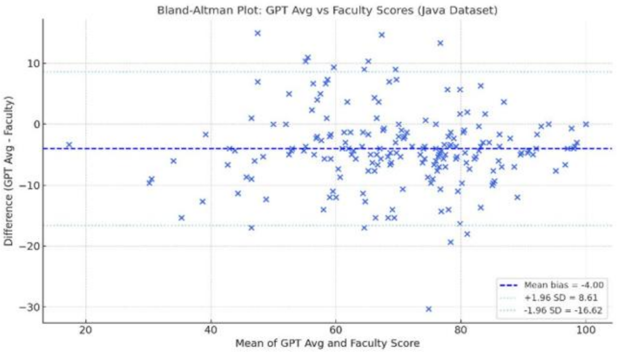


Fig. 5. Bland-Altman plot of GPT average vs. faculty scores (java dataset).

TABLE V. SAMPLE OF KOLMOGOROV-SMIRNOV TEST

	Python dataset	C++ dataset	Java dataset
GPT1	0.154	0.022	0.490
GPT2	0.127	0.001	0.516
GPT3	0.117	0.005	0.528
GPT average	0.128	0.003	0.607
Faculty	0.100	0.002	0.152

The results of the one-sample Kolmogorov–Smirnov (K-S) test revealed that the distributions of grades in the Python and Java datasets do not significantly deviate from normality, as all p-values were greater than 0.05 for both ChatGPT grading attempts and faculty-assigned scores. In contrast, the C++ dataset consistently showed p-values below the 0.05 threshold across all grading sources, indicating a significant departure from normality. Based on these findings, it is appropriate to use parametric tests, such as the Paired t-test, for statistical comparisons in the Python and Java datasets. However, for the C++ dataset, where normality is not satisfied, non-parametric test, specifically the Wilcoxon Signed-Rank test, is more suitable. This mixed-method approach ensures the statistical tests align with the underlying data distributions, thereby enhancing the validity of the comparative analysis.

Three statistical tests were used to compare the accuracy of grading by faculty and ChatGPT. The Two Sample Kolmogorov-Smirnov (K-S) Test was used to assess whether the distributions of faculty scores and GPT-generated scores are similar or differ significantly. In other words, it tests whether the two samples came from the same distribution or not. This test is particularly useful for understanding the overall behavior of the data without making strict assumptions about normality [27]. The following are the null and alternative hypotheses.

- Null Hypothesis (H0): The two samples (faculty scores and GPT scores) come from the same distribution.
- Alternative Hypothesis (H1): The two samples come from different distributions.

For the Two-Sample K-S test, p-values greater than 0.05 indicate no statistically significant difference between ChatGPT and faculty grade distributions, meaning they likely originate from the same population. Table VI shows the results of the test.

TABLE VI. TWO-SAMPLE K-S TEST

p-value	Python	C++	Java
GPT1	0.709	0.000	0.000
GPT2	0.425	0.001	0.011
GPT3	0.425	0.191	0.087
GPT average	0.425	0.000	0.006

For the Python dataset, GPT1 showed the highest p-value, suggesting its grades were most aligned with faculty assessments. In contrast, the lower p-values for GPT2, GPT3, and the average scores—particularly in the C++ and Java datasets, indicate a significant shift in distribution, reflecting a divergence from faculty grading. Despite this, the repeated results for GPT2 and GPT3 across datasets suggest consistent grading behavior by ChatGPT. The Two-Sample K-S test, when interpreted alongside the mean and correlation values, reveals key observations. In the Python dataset, GPT1 shows the closest mean (76.5) to faculty grading (77.6) and the highest K-S p-value (0.709), indicating strong similarity in distribution and grading behavior. In contrast, the C++ and Java datasets show lower GPT means compared to faculty, with significant K-S p-values (e.g., 0.000 for GPT1 in C++), suggesting ChatGPT graded more strictly in these datasets. However, the high Pearson correlations across all datasets (above 0.89) indicate

ChatGPT remains consistent with faculty in rank-ordering students' performance.

ChatGPT demonstrates strong consistency with faculty in grading trends (high correlations), but tends to assign lower grades, particularly in C++ and Java. The distributional differences highlighted by K-S tests suggest a systematic strictness in GPT grading, warranting rubric calibration to align mean grading behavior more closely with human assessors.

The Paired t-test was used to analyze the mean differences between the two sets of scores, assuming that the differences are normally distributed which was the case for the Python and Java datasets. This test quantified the extent to which the GPT model aligns with or deviates from faculty grading, providing a clear comparison of average performance. The following is the null and alternative hypotheses.

- Null Hypothesis (H0): The mean difference between the paired samples (faculty scores and GPT scores) is zero, indicating no significant difference in means.
- Alternative Hypothesis (H1): The mean difference between the paired samples is not zero, indicating a significant difference in means.

The Wilcoxon Signed-Rank test was used to assess whether there is a significant difference between two related samples. It is a non-parametric alternative to the Paired t-test and is especially appropriate when the data are not normally distributed. The test evaluates whether the median difference between paired observations is significantly different from zero. This made it suitable for comparing grading outcomes, particularly in the C++ dataset, where normality assumptions are violated. The following are the hypotheses:

- Null Hypothesis (H0): The median difference between the paired samples is zero.
- Alternative Hypothesis (H1): The median difference between the paired samples is not zero.

Table VII shows the results of the three tests to assess the statistical significance between GPT's grading versus faculty grading.

TABLE VII. STATISTICAL TESTS COMPARING CHATGPT AND FACULTY GRADES.

	Wilcoxon Test	Paired t-test	
	C++	Python	Java
	W-Statistic, *p*-value, Effect Size (*r*)	*t*-Statistics, *p*-value, Cohen's d	*t*-Statistics, *p*-value, Cohen's d
GPT1	1392, 0.00, 0.99	- 1.42, 0.15, -0.11	-12, 0.00, -0.84
GPT2	2680, 0.00, 0.57	- 4.10, 0.00, -0.34	-7, 0.00, -0.53
GPT3	3197, 0.00, 0.42	- 3.92, 0.00, -0.32	-5, 0.00, -0.38
GPT average	2488, 0.00, 0.99	- 3.11, 0.00, -0.25	-8, 0.00, -0.62

For the Python dataset, the Paired t-test for GPT1 resulted in a p-value of 0.157, which is greater than 0.05; therefore, we failed to reject the null hypothesis, indicating no significant difference in the mean of GPT1 and faculty grades. For GPT2, GPT3, and the average values, the p-values are less than 0.05, indicating a significant mean difference compared to faculty scores. The t-Statistics (e.g., -1.421 for GPT1) indicate the direction and magnitude of the difference. Negative values indicate that the faculty scores are higher than the GPT scores, which was also the case for the remaining results. For the Java dataset, all p-values are extremely low (0.000), and t-Statistics are large in magnitude, especially for GPT1 (t= -11.9), showing that GPT was much stricter than faculty, that is, ChatGPT consistently assigned lower grades than faculty in Java assignments. There is a clear pattern of stricter grading by ChatGPT across all attempts for Java. This may reflect the model's sensitivity to structure and modularity in Java code. While consistent, this strictness means ChatGPT should be used with human oversight for Java grading, and other programming assessments.

The Wilcoxon test over C++ dataset yielded p-values below 0.000, highlighting significant differences in the rank ordering of grades between ChatGPT and faculty. The consistent deviation in GPT scores suggests a more rigid and systematic grading pattern. These results confirm that ChatGPT's grading approach for C++ assignments differs notably from that of human evaluators. Although not reported, the t-statistic for the paired t-test on this dataset was negative, confirming that ChatGPT yielded lower grades on average and that this difference is statistically significant.

B. Research Question 2

1) RQ2: Evaluate the consistency of ChatGPT grading: To answer this question, descriptive statistics, intra-class correlation coefficients (ICC), and Two-Sample Kolmogorov-Smirnov (K-S) test were conducted to evaluate the consistency among GPT1, GPT2, and GPT3 for each dataset.

Across all three datasets, the means, standard deviations, and percentile values of GPT2 and GPT3 were remarkably close, indicating a high degree of internal consistency between repeated grading attempts. For instance, in the Python dataset, GPT2 and GPT3 both had a mean of 75.1 and identical standard deviations (17.10), with overlapping interquartile ranges. The first attempt (GPT1), while still comparable, showed slightly higher variation and mean, suggesting that ChatGPT's initial grading could occasionally deviate before stabilizing in later attempts. Similarly, in the C++ dataset, there was a progressive increase in mean from GPT1 (65.6) to GPT3 (70.4), which may indicate some degree of refinement across attempts. Yet, the standard deviation decreases slightly, suggesting a more concentrated distribution and reinforcing grading stability over time. In the Java dataset, GPT2 and GPT3 again showed high consistency, with means of 68.4 and 69.3, and virtually identical standard deviations (15.1–15.3). GPT1, while still within a close range, had a slightly lower mean (65.9) and similar spread.

Combining the analysis across the three datasets (Python, C++, and Java) showed that ChatGPT demonstrates a high degree of consistency, reproducibility, and stability in its grading behavior. The descriptive statistics for GPT1, GPT2, and GPT3 across all datasets revealed closely aligned mean values, standard deviations, and quartile ranges. This alignment indicated reliable and uniform grading patterns across multiple attempts and programming languages. In all cases, GPT1 tended to deviate slightly from GPT2 and GPT3, suggesting that the first attempt may reflect more variability, while the subsequent attempts showed improved alignment. The narrow spread of scores across the grading attempts and their consistent distributions confirmed that ChatGPT provides dependable grading performance, regardless of language or context. These patterns reinforce its suitability as an automated assessment tool, particularly when used with clear rubrics and repeated grading to enhance robustness. Fig. 6 shows the boxplot distribution of the three measures in the datasets. The figure shows the closeness between the values, mean, min, max, and the quartiles. This demonstrates consistency in grading for the three attempts with a potential variability in the first attempt.

The ICC is a statistical measure used to assess the reliability or consistency of measurements made by different raters or across different conditions [28]. It is particularly useful in scenarios where multiple observations or measurements are made under similar circumstances which applies in this study. The Single rater/ measurement [ICC (1)] has been used to measure the consistency across multiple subjects for one rater as recommended in (OpenAI, 2025). The ICC measured how consistent the ChatGPT was in grading across the three different grading attempts. A high ICC indicates that the model provides stable and consistent grades across attempts, which would suggest strong consistency. ICC value ranges from 0 to 1, where 0 indicates no consistency and 1 indicates perfect consistency. The ICC is computed as a ratio of the variance between groups to the total variance, which includes the variance within groups [28]. Table VIII shows the ICC results for the three GPT attempts. The ICC1 measures the absolute agreement among raters when considering each rater's scores independently. The ICC2 measures the consistency of ratings assuming that raters are randomly selected from a larger population. The ICC3 measures the consistency of ratings under fixed conditions, assuming the same raters are always used.

The Python dataset showed excellent grading consistency (ICC = 0.981–0.984, p = 0.000), with tight confidence intervals (0.97–0.99), indicating high precision and reproducibility. The C++ dataset also demonstrated strong reliability (ICC = 0.948–0.968), with slightly greater variability under random rater assumptions. Similarly, the Java dataset achieved high consistency (ICC = 0.961–0.974, p = 0.000), confirming strong agreement across grading attempts. Overall, ChatGPT maintained very good to excellent grading stability across all datasets, with slightly improved consistency under fixed-rater models.

Additionally, the Two-Sample Kolmogorov–Smirnov (K–S) test was applied to compare score distributions across grading attempts, evaluating whether they originate from the same underlying distribution based on the K–S statistic and corresponding p-values, as shown in Table IX.

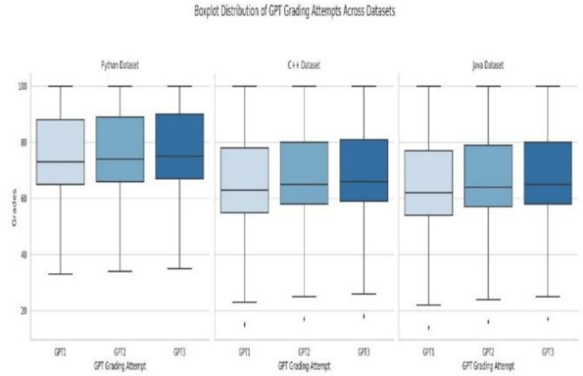


Fig. 6. Boxplot distribution of GPT1, GPT2, and GPT3.

TABLE VIII. THE ICC TEST RESULTS

Python								
	Type	Description	ICC	F	df1	df2	pval	CI95%
1	ICC1	Single raters absolute Agreement	0.981	163	145	292	0.000	[0.98, 0.99]
2	ICC2	Single random raters Consistency	0.981	187	145	290	0.000	[0.97, 0.99]
3	ICC3	Single fixed raters Consistency	0.984	187	145	290	0.000	[0.98, 0.99]
C++								
1	ICC1	Single raters absolute Agreement	0.948	56	187	376	0.000	[0.93, 0.96]
2	ICC2	Single random raters Consistency	0.948	93	187	374	0.000	[0.87, 0.97]
3	ICC3	Single fixed raters Consistency	0.968	93	187	374	0.000	[0.96, 0.98]
Java								
1	ICC1	Single raters absolute Agreement	0.961	76	199	400	0.000	[0.95, 0.97]
2	ICC2	Single random raters Consistency	0.961	116	199	398	0.000	[0.91, 0.98]
3	ICC3	Single fixed raters Consistency	0.974	116	199	398	0.000	[0.97, 0.98]

TABLE IX. TWO-SAMPLE K-S TEST RESULTS

	Python dataset		C++ dataset		Java dataset	
Pairs	Statistics	P-value	Statistics	P-value	Statistics	P-value
GPT1 vs. GPT2	0.082	0.709	0.127	0.093	0.080	0.545
GPT1 vs. GPT3	0.075	0.803	0.223	0.000	0.110	0.177
GPT2 vs. GPT3	0.006	0.999	0.106	0.238	0.045	0.987

The Kolmogorov–Smirnov (KS) test results for the Python dataset demonstrate strong consistency across ChatGPT's three grading attempts. All p-values are well above the 0.05 threshold, indicating no statistically significant differences in the distributions between GPT1, GPT2, and GPT3. The comparison between GPT2 and GPT3 yields a particularly high p-value of 0.999, suggesting near-perfect alignment. These results affirm that ChatGPT's grading behavior for Python programming assignments is stable, reproducible, and consistent across multiple attempts.

In the C++ dataset, the K-S test showed mixed results. While GPT1 vs. GPT2 (p = 0.093) and GPT2 vs. GPT3 (p = 0.238) exhibit no significant distributional differences, the comparison between GPT1 and GPT3 revealed a statistically significant p-value of 0.000. This suggests a deviation in grading behavior between the first and third attempts, indicating that ChatGPT may produce slightly more variable outputs in C++ contexts. This inconsistency highlights the need for closer calibration when using ChatGPT in such environments. The K-S test results for the Java dataset reflected strong consistency among all grading attempts. None of the p-values indicate significant differences, and the GPT2 vs. GPT3 comparison yields a very high p-value of 0.987, pointing to excellent alignment. While GPT1 vs. GPT3 presents a somewhat lower p-value (0.177), it still remains non-significant. These findings suggest that ChatGPT maintains a reliable grading pattern when assessing Java programming assignments, supporting its suitability for consistent and repeatable evaluations in this context.

The empirical results for RQ1 and RQ2 suggest that the second and third grading attempts showed closer alignment with faculty grading compared to the first attempt. This convergence may be influenced by factors related to ChatGPT's underlying mechanisms and internal processing algorithms and strategies [29] as follows:

2) Stochastic variability in initial attempts: ChatGPT, like other LLMs, operates probabilistically. The first grading output may introduce slight randomness or conservative interpretation when processing the rubric and student code. As a result, GPT1 might diverge more from human reasoning, especially in edge cases or non-standard solutions. In contrast, repeated prompts tend to "smooth out" these stochastic effects.

3) Internalization of prompt structure and rubric: Repeated grading prompts, even when run separately, benefit from the model's ability to better adhere to structured expectations over multiple generations. GPT2 and GPT3 likely exhibit improved understanding and consistent application of rubric criteria, resulting in more rubric-aligned judgments and reduced deviation from faculty grading.

4) Model behavior in structured evaluation tasks: LLMs tend to perform better in well-structured, rule-driven tasks after multiple iterations. The clarity of the rubric, combined with repeated use of the same grading instructions, reduces ambiguity in scoring and improves alignment. GPT2 and GPT3 may represent more "settled" responses, less influenced by uncertainty or misinterpretation.

5) Internal consistency through pattern recognition: Although ChatGPT does not learn from previous grading attempts, its high pattern recognition capability enables it to detect recurring student mistakes and rubric violations more reliably across multiple runs. This cognitive consistency, manifesting in GPT2 and GPT3. reflects its ability to refine alignment with rubric criteria and faculty standards through internal convergence rather than experiential learning.

C. Research Question 3

1) RQ3: Assess ChatGPT's adherence to grading instructions and rubrics: Goal 3 was evaluated subjectively by grading the same assignment with different rubrics to test grading correctness.

The rubric submitted to the ChatGPT must be clearly defined and aligned with the targeted learning outcomes, course level, and expectations. The criteria that could be used to assess a programming assignment vary. For example, a programming assignment can be graded based on: problem understanding, correctness, completeness, code efficiency, error handling, modularity, code readability, adherence to standards, documentation and comments, code reusability, use of data structures, testing and debugging, creativity and innovation, scalability, version control, team collaboration, resource management, integration of external libraries, dynamic behavior, security considerations, optimization, and many more. Thus, the rubric should be carefully designed as per the expected outcome.

Table X shows four different rubrics that were used to grade the model answer of Fibonacci solution in Section III provided by the author of [26].

Model Answer:

```
## Compute and display the Fibonacci number requested by the user.
## Read input from the user.
n = int(input("Enter an integer: "))
# Handle the special cases for the first two Fibonacci numbers.
if n == 1 :
  fnew = 1
elif n == 2 :
  fnew = 1
else :
  # Compute the Fibonacci number.
  fold1 = 1
  fold2 = 1
  for i in range(2, n) :
    fnew = fold1 + fold2
    fold1 = fold2
    fold2 = fnew
# Display the Result.
print("Fibonacci number", n, "is", fnew)
```

TABLE X. RUBRIC SENSITIVITY DEMONSTRATION

	Criteria	Rubric description (Above Expectations) scale	Generated Results/ feedback
Rubric 1	Completeness	Implements the Fibonacci algorithm correctly, prompts the user for input, and calculates the nth Fibonacci number without errors; handles edge cases (e.g., n = 1 or n = 2).	4/4
	Correctness	Produces correct results for all test cases, including edge cases (e.g., n = 1, 2, or large n); follows the specified algorithm precisely.	4/4
Rubric 2	Completeness	Implements the Fibonacci algorithm correctly, prompts the user for input, and calculates the nth Fibonacci number without errors; handles edge cases (e.g., n = 1 or n = 2).	4/4
	Modularity	Code is highly modular, using well-structured functions or classes to separate logical tasks clearly.	2/4 The code lacks modularity. There is no use of functions or classes to organize the program. The entire logic is implemented in a single block.
Rubric 3	Code efficiency	Code is highly optimized, with minimal time and space complexity; avoids unnecessary operations.	¾ The code works efficiently for smaller inputs but could be improved for larger values of n.
	Code reusability	Code is modular and easily reusable, with clear functions or classes for different tasks.	2/4 The code lacks modularity and reusability. It implements all logic in a single block rather than breaking it down into reusable functions or classes.
Rubric 4	Error handling	Implements comprehensive error handling, catching all possible errors, including invalid input, edge cases, and unexpected exceptions.	1/4 The solution does not include any error handling. It does not validate user input.
	Use of data structures	Selects and uses appropriate data structures (e.g., lists, dictionaries, sets) to optimize problem-solving and algorithm efficiency.	2/4 The solution uses simple variables (fold1, fold2, fnew) to calculate the Fibonacci sequence. These are basic variables, not data structures like lists, tuples, or dictionaries. The program doesn't take advantage of more advanced data structures to optimize the solution or improve flexibility. For example, using a list could simplify storing Fibonacci numbers for later use or for a more generalized approach.
Rubric-free (prompt)	No criteria were given, but a simple prompt was provided: "*Grade the following student code out of 8 marks. provide feedback.*"	In the absence of predefined grading criteria, ChatGPT inferred and applied the following evaluation criteria based on the prompt: correctness of logic, use of conditionals, use of repetition, modularity, and input / output usability.	7/8 Strengths: Correct implementation of Fibonacci logic. Handles edge cases properly (n = 1 and n = 2). Code is readable and easy to follow. Proper use of control structures (if, for). Weaknesses: Lack of modularity: The code does not use functions, which limits reusability and clarity.

Each rubric evaluated different dimensions of the same "model answer" code solution: functional correctness, code structure, efficiency, error handling, and use of appropriate tools (like data structures). This variation highlights that ChatGPT's grading accuracy is directly influenced by the specificity, rubric's criteria, and grading instructions. This demonstrates the subjectivity of grading when different aspects are prioritized. It also shows that ChatGPT can be an effective grader when clear criteria are used. Considering the provided feedback, ChatGPT was able to distinguish and assess the model answer differently for each rubric.

The variation in grades across rubrics highlights an important point about the objectivity and subjectivity inherent in automated grading. While a single rubric might prioritize correctness and completeness, others might focus on factors like modularity, efficiency, or error handling. In the absence of a predefined rubric, ChatGPT autonomously formulates its own evaluation criteria based on the prompt and assesses the assignment accordingly.

The results coincide with what is noted in assessment theory, the rubric's design directly affects the final grade. Therefore, a low grade in one rubric does not necessarily reflect a failure in the solution, but rather an area that was not emphasized by the rubric [30]. Considering these results, the rubric's criteria should be carefully selected based on the course level, expectations, and learning outcomes. Therefore, we propose a practical and flexible framework that enables collaborative rubric development between instructors and any LLM as ChatGPT to support any programming criteria or dimensions mentioned above.

2) Instructor–AI collaborative rubric development framework: The objective of this framework is to transform LLMs, such as ChatGPT, from passive tools into active educational collaborators. By doing so, instructors can scale high-quality, rubric-aligned assessments while maintaining academic rigor, fairness, and transparency. This adaptable framework supports a wide range of evaluation dimensions relevant to programming assignments, regardless of the programming language. The following outlines a step-by-step approach to develop an instructor–AI collaborative rubric framework for programming evaluation:

a) Learning outcome alignment: Instructors define the learning outcomes based on the course level and the targeted assessment objectives, such as problem-solving, algorithmic efficiency, and documentation.

b) Define core dimensions: Instructors select relevant criteria from a broad repository, i.e., (Correctness, code efficiency, modularity, error handling, documentation, reliability...,etc.) or define new ones. Each criterion is described at multiple students' performance levels (Above expectations, within expectations, approach expectations, below expectations).

c) Weight assignment and scaling: Each criterion is assigned a weight based on its importance in the given task.

d) Prompt engineering and rubric encoding: Craft precise prompts to obtain rubric-aligned responses from LLMs. In this study, a structured command embedding the rubric criteria, weights, and expectations was consistently used along with the below command and the assignment to be graded. This ensures systematic, transparent evaluation aligned with instructional standards, reinforcing fairness and consistency.

e) Instructor review and calibration: Instructor review plays a vital role in this phase, ensuring the accuracy and reliability of the AI-generated grades. This review involves comparing the LLM's outputs with human-assigned scores to detect any inaccuracies, inconsistencies, or feedback hallucinations. Based on the findings, instructors can refine rubric descriptions, adjust weights, and improve prompt clarity to enhance grading alignment and reduce potential errors in future assessments.

f) Iterative improvement and feedback loop: Finally, both AI prompts and rubric structures should be iteratively refined over time based on student feedback, identified grading anomalies, and evolving course objectives. This continuous improvement ensures that the AI remains aligned with instructional goals, adapts to pedagogical shifts, and maintains fairness and clarity in the grading process.

By following this framework, instructors ensure that grading is rubric-driven, grounded in clearly defined instructional criteria, and guided by learning outcomes. It enhances transparency by embedding explicit scoring rationales, reducing concerns about AI as a "black box". The approach maintains a strong pedagogical focus, as feedback is directly mapped to learning goals, highlighting both strengths and areas for improvement. Finally, it promotes bias mitigation by requiring all deductions to be justified through rubric-based citations minimizing subjective grading and reinforcing fairness.

D. Research Question 4

1) RQ4: Gather qualitative feedback from students about ChatGPT grading. Gathering students' perceptions about the effectiveness, fairness, and usability of ChatGPT when assessing programming assignments is crucial. Students are directly affected by the grading process, and may have perspectives on how well ChatGPT's feedback aligns with their understanding of the rubrics and the assignment requirements.

To assess students' perceptions, they received their graded assessments, and their feedback was collected to evaluate grading correctness, clarity, fairness, trust, transparency, and the usefulness of feedback. Finally, after students became aware of their grades, they were informed that the grading had been performed automatically using ChatGPT.

TABLE XI. PRE-REVEAL AND POST-REVEAL SURVEY QUESTIONS

Survey Questions: Pre-Reveal	Survey Questions: Post-Reveal
General Perception of Grading How satisfied are you with the grading process for this assignment? [1, 2, 3, 4, 5] How fair do you think the grading was? [1, 2, 3, 4, 5] How clear and helpful was the feedback you received? [1, 2, 3, 4, 5]	**General Perception of Grading** How satisfied are you with ChatGPT's grading process for this assignment? [1, 2, 3, 4, 5] How fair do you think ChatGPT's grading was? [1, 2, 3, 4, 5] How clear and helpful was the feedback ChatGPT provided? [1, 2, 3, 4, 5]
Trust in Grading Do you trust the grading process to evaluate your work accurately? [1, 2, 3, 4, 5] Do you think the grading is consistent across all students? [1, 2, 3, 4, 5]	**Trust in Grading** Do you trust ChatGPT to evaluate your work accurately? [1, 2, 3, 4, 5] Do you think ChatGPT's grading is consistent across all students? [1, 2, 3, 4, 5]
Transparency in Grading I understand how my programming assignment was evaluated against the rubric criteria. [1, 2, 3, 4, 5]	I understand how my programming assignment was evaluated against the rubric criteria. [1, 2, 3, 4, 5]
Usefulness of Feedback How useful was the feedback in helping you understand the assignment's requirements? [1, 2, 3, 4, 5] What did you like most about the feedback you received? (Open-ended) What aspects of the grading process would you improve? (Open-ended) Any additional remarks you would like to highlight?	**Usefulness of Feedback** How useful was ChatGPT's feedback in helping you understand the assignment's requirements? [1, 2, 3, 4, 5] What did you like most about ChatGPT's feedback? (Open-ended) What aspects of ChatGPT's grading process would you improve? (Open-ended) Any additional remarks you would like to highlight? Did you feel ChatGPT was more or less lenient than a human instructor? Please explain your perception. In your opinion, what role should instructors play when AI is used to grade programming assignments? Have you ever received feedback from ChatGPT that felt incorrect, unfair, or overly harsh? If yes, please describe. How would you compare the quality of feedback from ChatGPT to that of a human instructor?

To achieve this, two surveys were conducted: one prior to informing students that their assessments were graded using ChatGPT, and another afterward. The questions for both surveys are presented in Table XI. Both surveys utilized a 5-point Likert scale, enabling participants to articulate their level of agreement or perception effectively. The scale ranged from one extreme to another, such as: 1. Very Unfair, 2. Somewhat Unfair, 3. Neutral, 4. Somewhat Fair, 5. Very Fair. This approach ensured a structured method for capturing nuanced feedback while maintaining consistency and comparability across responses. The survey was administered during the Fall and Spring semesters of the 2024/2025 academic year across three academic institutions, targeting students enrolled in Introduction to Programming and OOP courses. Participation was voluntary, and students were assured of both anonymity and confidentiality. A total of 158 students participated in the survey. The results of the pre-reveal and post-reveal surveys are shown in Fig. 7 and Fig. 8, respectively.

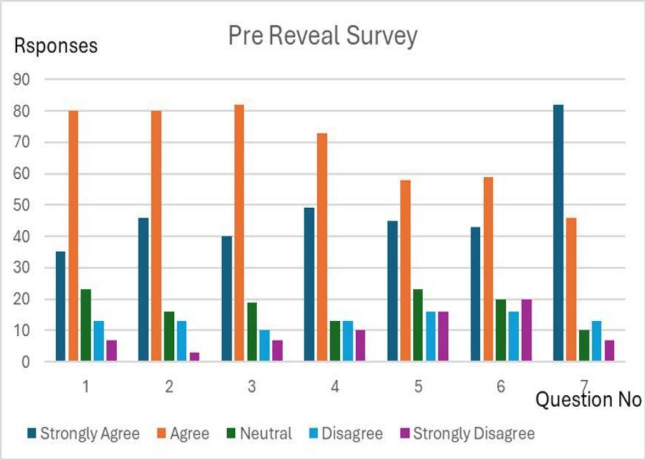


Fig. 7. Pre-reveal survey results.

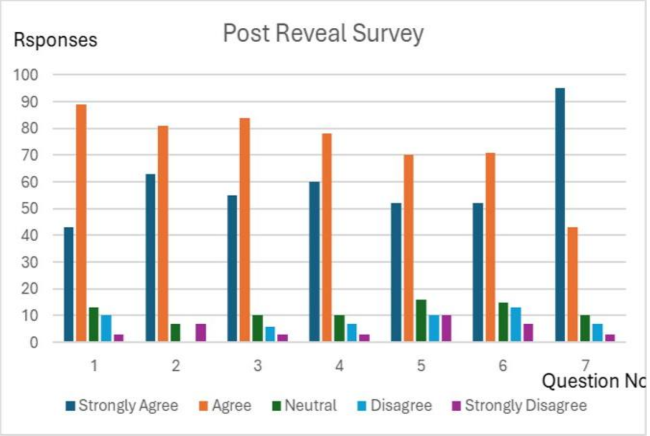


Fig. 8. Post-reveal survey results.

The pre-reveal survey results suggest that students are generally satisfied with the grading process, perceive it as fair, and find the feedback useful for understanding assignment requirements. However, there are areas where students express some concerns, particularly regarding grading consistency, fairness, and understanding of the rubric.

These results provide a baseline for comparison with post-reveal perceptions, especially as students learn about the use of ChatGPT in grading. The concerns about fairness and understanding suggest that, even with ChatGPT's involvement, addressing transparency, consistency, and communication could further enhance students' trust and satisfaction with the grading process.

After learning that ChatGPT was used as the grading tool, students' satisfaction, trust, and perception of fairness all improved. Most notably, the trust in grading accuracy and consistency saw significant increases. This suggests that students felt more confident in the consistency and fairness of the grading process once they knew an AI was involved. Feedback clarity and usefulness remain consistent between the pre-reveal and post-reveal surveys, with students continuing to value the feedback they received, regardless of whether it came from ChatGPT or a human grader. There was a notable improvement in the understanding of the grading rubric in the post-reveal survey, indicating that students may appreciate the transparent and rule-based nature of ChatGPT’s grading. The post-reveal survey results indicated that students generally had a positive response to ChatGPT being used as a grading tool. The positive shift in perceptions emphasizes the potential of using ChatGPT as a grading tool to enhance consistency, fairness, and transparency. It also pointed to the importance of clearly communicating the grading process and providing detailed feedback, whether the grading is done by a human or an AI system.

Students expressed notable interest in utilizing ChatGPT as a prior-submission assessment tool to optimize their grades. They believed it could serve as a valuable resource to identify and address potential issues before final submission. However, implementing such a tool in formal educational systems would necessitate significant alterations to traditional assessment methodologies, a transformative change that extends beyond the scope of this study. Students expressed a preference for ChatGPT-based grading, citing several key advantages: it eliminates human error and bias, ensuring a greater level of consistency. They appreciated the speed and promptness of receiving comprehensive feedback, as well as the interactive environment that fosters conducive learning. Additionally, ChatGPT provides rich, detailed insights into their work and clear guidance for improvement, enhancing their overall learning experience. Conversely, student responses revealed nuanced perspectives on ChatGPT’s grading. While many appreciated its objectivity, clarity, and speed, several expressed a preference for human grading due to perceived leniency, especially when dealing with partially correct or unconventional solutions. Some raised concerns about ChatGPT being overly strict with edge cases or non-optimal implementations. A common suggestion was to adopt a hybrid approach, that is, using ChatGPT for initial evaluation, followed by instructor review to ensure context-aware and empathetic grading.

To statistically validate the reported results, the non-parametric Wilcoxon Signed-Rank test was employed to determine whether there was a statistically significant difference between the pre-reveal and post-reveal survey responses. This test is particularly suitable as it does not assume normal distribution of the data. Additionally, the effect size (r) was calculated to assess the magnitude of the observed difference, providing a complementary perspective to the statistical significance. Fig. 9 shows the weighted average for students' responses for each question.

Wilcoxon Statistic: W = 0.0. This result showed that all post-reveal scores were consistently higher than their corresponding pre-reveal scores, indicating a clear directional change in the data.

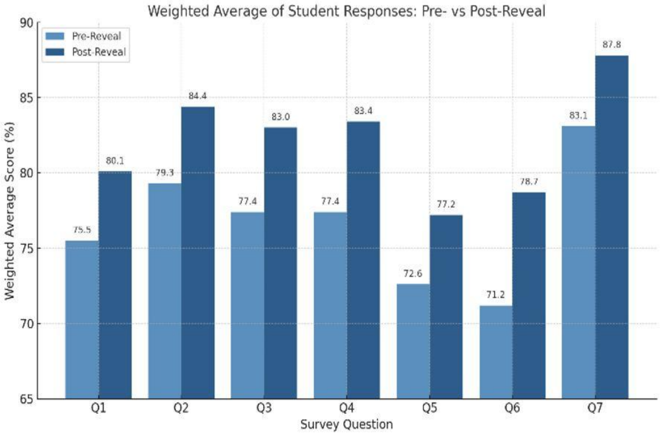


Fig. 9. Weighted average of student's responses.

P-value = 0.015, which is below the standard threshold of 0.05, indicated a statistically significant difference. This means the difference between pre-reveal and post-reveal scores is unlikely due to chance. The estimated Z-score (-2.417) shows a shift toward higher post-reveal scores. The effect size (r = 0.913) reflects a strong relationship, yet the overall change is not substantial enough to have meaningful educational or operational significance. The analysis of responses from 158 students strongly supports the conclusion that revealing the use of ChatGPT in grading had a positive influence on student perceptions. Specifically:

- Trust in the grading process improved significantly, with students frequently attributing this to ChatGPT's perceived objectivity, consistency, and transparency qualities that many felt made it more systematic than human evaluation.
- Satisfaction levels increased, often linked to the clarity, structure, and depth of ChatGPT's feedback, which helped students better understand their mistakes and areas for improvement.
- While some students viewed ChatGPT as more strict than human instructors, many still appreciated its rigor, considering it fair and unbiased.
- Notably, many students supported a hybrid approach, recommending that AI grading be supplemented by instructor oversight to ensure contextual understanding, empathy, and alignment with educational intent.

Moreover, the consistency of positive shifts across all survey items suggests a broad and uniform improvement in perception, rather than isolated changes, further reinforcing the students' growing acceptance of ChatGPT as a supportive grading tool when transparently and responsibly integrated.

V. Considerations, Recommendations and Limitations

This section summarizes the key considerations, actionable recommendations, and inherent limitations of integrating AI-based tools such as ChatGPT into academic assessment practices. To ensure responsible adoption, educators and institutions must address pedagogical, ethical, and operational factors based on the empirical findings of this study. While AI-assisted grading offers significant advantages in terms of efficiency, consistency, and scalability, its effectiveness heavily depends on rubric quality, human oversight, and a ready institution. Furthermore, this section acknowledges that generalizability and fairness may be limited by technical, ethical, and contextual constraints. With these insights, educators, researchers, and policymakers will be able to use AI in educational assessment in an informed, transparent, and sustainable manner.

A. Considerations

The integration of AI tools like ChatGPT into grading assignments presents a paradigm shift in educational assessment. While these tools offer advantages such as objectivity, speed, and consistency, they also raise ethical, practical, and pedagogical challenges. To ensure the effective adoption and sustainable use of AI grading systems, this section outlines key guidelines, considerations, and actionable recommendations based on the findings and experiences of this study. Educators who are considering using ChatGPT as an assessment tool for grading programming assignments should be mindful of several key considerations. The following summarizes the considerations that an evaluator should consider when using ChatGPT as an assessment tool:

1) Ensure clear and specific rubrics: The study highlights how different rubrics yield different grades for the same solution, indicating that the grading criteria directly impact the final score. Therefore, educators should ensure that their rubrics are well-defined and explicitly aligned with the learning objectives of the course. For example, if the rubric includes criteria such as modularity or error handling, the solution's grade will depend on how closely the solution adheres to those standards, even if the solution is functionally correct and complete as per the question requirements. Clear rubrics will help reduce ambiguity in grading and guide the AI tool to apply consistent standards.

2) Clarify expectations for each criterion: Given that ChatGPT's grading performance varies with different criteria (e.g., modularity, efficiency, and error handling), it is essential to explain to students how their solutions will be assessed under these different criteria. This will help students understand that one rubric may reward correctness while another may place more emphasis on aspects like code organization or optimization. Educators should also ensure that students understand the grading rationale behind each rubric to avoid confusion when the final grade is lower under certain criteria.

3) Balance between human and AI grading: While ChatGPT can effectively grade programming assignments based on specific rubrics, human oversight is still necessary to ensure fairness and consistency. Educators should treat ChatGPT's grades as part of a blended assessment model, where the AI's grading is complemented by human judgment. This hybrid approach would allow educators to consider subjective aspects of programming, like creativity or logical thinking, which may be harder for ChatGPT to assess. For example, feedback on code structure or how well the solution meets the real-world problem could be better evaluated by a human.

4) Training ChatGPT on domain-specific rubrics: For educators to make ChatGPT a more reliable and precise grader, it might be beneficial to fine-tune the model on domain-specific rubrics and grading patterns. While ChatGPT has a broad knowledge base, its effectiveness as an assessor could be improved by customizing it to specific course objectives or standards. This was clearly demonstrated after the second and third grading attempts, showing negligible differences.

5) Monitor and review AI grading results: As with any automated system, it's important for educators to monitor the grading outcomes when using ChatGPT to ensure the tool is producing fair and reliable results. If discrepancies between ChatGPT's grades and human assessments are observed, the rubrics or model settings should be revised. Educators should maintain an iterative review process, continuously adjusting the system for improved alignment with their grading standards.

6) Computational cost, scalability, and accessibility challenges: ChatGPT grading was conducted using OpenAI's API, where each grading session (including rubric parsing and code evaluation) required approximately 15-20 seconds per assignment, on average. Compared to manual grading, which can take 5-10 minutes per submission, this represents a reduction of over 90% in grading time. Given that the study processed 534 programming assignments, the total automated grading time was approximately 2.5 hours versus an estimated 45+ hours of human effort. At current API pricing ($0.03–$0.06 per 1K tokens), the average cost per graded assignment remained under $0.05 USD. This demonstrates that ChatGPT-based grading is both time-efficient and cost-effective, particularly for large-scale courses.

Institutions may face limitations related to service availability, as AI platforms can occasionally experience disruptions or access issues. In parallel, accessibility remains a critical concern. Institutions in under-resourced or developing regions may lack the necessary infrastructure, internet bandwidth, or financial capacity to integrate such AI tools effectively. This could further widen the digital divide and create inequities in the adoption of AI-enhanced educational technologies. To address these challenges, this study emphasizes the importance of exploring cost-effective and accessible alternatives, including open-source and lightweight LLMs (e.g., DeepSeek, TinyLlama), as well as options for localized or offline deployment. Such strategies could significantly improve scalability, reduce dependency on commercial platforms, and ensure more equitable access to AI-driven assessment tools across diverse educational contexts.

7) Integration of AI in academia: The successful integration of AI technologies, such as ChatGPT, into academic workflows goes beyond technological readiness, requiring alignment at institutional, cultural, and pedagogical levels. While this study demonstrates ChatGPT's potential to enhance grading consistency, efficiency, and objectivity when supported by well-structured rubrics and human oversight, achieving these benefits at scale demands deliberate institutional commitment and coordination.

Academic institutions may face several integration challenges. These include concerns over the reliability and interpretability of AI-generated assessments, the risk of over-reliance on automated tools, and the need to preserve faculty autonomy and academic rigor. Furthermore, the introduction of AI tools in assessment practices may require significant shifts in instructional strategies, grading policies, and curriculum design. To support responsible and effective AI adoption, institutions must invest in comprehensive faculty development initiatives, promote transparency in AI usage, and establish governance frameworks that clarify accountability and ethical boundaries. Engagement from all academic stakeholders such as faculty, administrators, students, and IT departments is essential to foster trust and build institutional capacity. Pilot programs, feedback loops, and iterative evaluation processes can serve as stepping stones toward wider acceptance and long-term sustainability.

B. Recommendations

The integration of AI tools into educational assessment presents both opportunities and ethical responsibilities. To uphold academic integrity, fairness, and student trust, this study proactively implemented safeguards across four core pillars: data anonymization, privacy compliance, bias mitigation, and educator accountability. Recognizing that the use of AI in grading must be both effective and ethically sound, the following recommendations and precautions are proposed. These guidelines aim to support institutions and educators in responsibly leveraging AI technologies to enhance grading transparency, consistency, and efficiency while also mitigating risks related to privacy, bias, and over-reliance on automated systems.

1) Human oversight and ethical use in AI grading: In AI-assisted grading, maintaining transparency and ensuring robust human oversight are essential ethical practices. Faculty must retain full responsibility for final grades. Their expertise is vital for interpreting nuanced aspects of student work, such as creativity, intent, or alignment with course objectives that AI may overlook. While AI tools like ChatGPT offer efficiency and consistency, they are still subject to limitations including hallucinations, uncertainty, and evolving behavior. To mitigate these risks, instructors are empowered to review, adjust, or override AI-generated evaluations. Furthermore, students should be provided with the opportunity to appeal or challenge automated grades, fostering an inclusive, dialogic assessment environment. Equitable access to AI-supported learning tools must also be prioritized, particularly for students with diverse learning needs or disabilities. This blended approach ensures that AI complements, rather than replaces, the human judgment necessary for fair and academically sound evaluation.

2) Bias mitigation strategies: To address the possibility of algorithmic bias [31], the study implemented several safeguards. A rubric-driven framework was used to ensure standardized and objective grading criteria, reducing the likelihood of inconsistent or arbitrary evaluation. Pre-grading checks were performed to catch skewed deductions, especially in cases involving unconventional yet valid solutions. Moreover, a diverse sample of student submissions, spanning varying performance levels, was selected to limit the effect of training data bias. All grading outputs were then reviewed by human instructors to verify their accuracy and appropriateness, reinforcing fairness in judgment.

3) Ensure transparency and communication: Institutions should ensure transparent communication with students and faculty regarding the use of AI in grading. Providing clear, detailed information about the role of AI, the grading process, evaluation criteria, and known limitations fosters trust and minimizes scepticism. Additionally, academic policies and procedures should be updated to reflect the evolving landscape brought about by AI technologies. These updates should address critical aspects of responsibility and accountability, ensuring alignment with institutional values while accommodating the integration of AI-driven tools like ChatGPT.

4) Data privacy, anonymization, and compliance in AI grading: To ensure ethical AI usage [32], this study prioritized robust data privacy practices. All student submissions were pseudonymized, removing identifiable information such as names or student IDs before being processed by ChatGPT. The grading process adhered to institutional IRB protocols and complied with international data protection regulations like GDPR and FERPA. Using OpenAI's enterprise API ensured that no user data was stored or reused post-processing, minimizing the risk of data leakage.

Despite AI providers offering secure communication channels with end-to-end encryption and access controls, users still bear responsibility for avoiding the submission of sensitive personal data. For example, Italy fined OpenAI 15 million euros after an investigation revealed that the company had processed users' personal data to "train ChatGPT without having an adequate legal basis and violated the principle of transparency and the related information obligations towards users"[31].

5) Collecting continuous feedback: Finally, institutions should routinely assess the effectiveness and fairness of AI tools. Feedback from both students and faculty should guide continuous improvements in these systems.

C. Limitations

Educators should be aware of the limitations of AI-based grading [33]. ChatGPT, despite its capabilities, may struggle with understanding complex human creativity in coding or with solutions that involve non-standard problem-solving approaches. These elements might not always be captured adequately in a rubric, so human grading remains necessary to address those nuances. The tool is very beneficial, but there are several domains where it is inefficient. Here are the key limitations associated with ChatGPT and other LLMs:

1) Scope and generalizability: Despite involving datasets from three programming languages (Python, C++, and Java) and multiple academic institutions, the study's scope remains relatively narrow. The findings may not fully generalize to broader educational contexts such as non-STEM disciplines, advanced-level programming courses, or diverse cultural and pedagogical environments. For example, rubrics may not account for regional educational norms (e.g., coding styles favored in India vs. the U.S.). Expanding future work to include varied curricula and international samples would enhance external validity.

2) Dependence on rubric quality: ChatGPT's grading effectiveness hinges on the clarity, structure, and alignment of the input rubric with the intended learning outcomes. Ambiguities, poorly defined criteria, or lack of granularity in performance descriptors can lead to inconsistent assessments. Instructors must therefore invest time in carefully designing rubrics to ensure that AI-generated grades are both meaningful and accurate.

3) Opaqueness of AI decision-making (black-box problem): As with most large language models, ChatGPT operates as a black-box system, meaning that its internal reasoning processes are not directly observable. While the outputs appear coherent and justifiable, there is no clear mechanism to audit or explain how specific grades were derived. This limits trust and interpretability, especially in high-stakes educational settings. A potential way to overcome the black-box limitation is through Chain-of-Thought (CoT) [34], [35] prompting, a prompt-engineering technique that guides large language models to articulate their intermediate reasoning steps before producing a final decision. By exposing these logical steps, CoT prompting helps clarify how the model arrives at its conclusions, making its decision-making process more interpretable and auditable. In the context of AI-based grading, CoT prompting can help break the opacity of the model's reasoning, explain how rubric criteria are applied, justify assigned grades, and improve reproducibility across repeated grading sessions.

4) Use of static prompting: This study utilized fixed, pre-defined prompts to ensure uniformity in the grading process across assignments. However, such a static approach may restrict the model's responsiveness to subtle differences in task context. Future work could explore dynamic or adaptive prompting strategies that adjust based on assignment type, student level, or rubric complexity, potentially improving alignment with human grading standards.

5) Faculty grading as an imperfect benchmark: Faculty-assigned grades served as the ground truth in this evaluation. However, human grading is inherently subjective and prone to inconsistency or error. Variations in interpretation, leniency, and familiarity with the student can all affect grading reliability. This introduces a limitation in using faculty scores as the sole comparator when assessing ChatGPT's "correctness". Future research will involve having each assignment graded independently by two faculty members.

6) Model and environmental variability: ChatGPTs outputs may be influenced by several external factors, including server load, temperature settings, and ongoing backend updates by OpenAI. Even with the same prompt and input, repeated grading sessions may yield slightly different outputs over time. Furthermore, changes between different versions of ChatGPT (e.g., GPT-3.5 vs GPT-4) can impact reproducibility, posing challenges for longitudinal studies and standardization in AI-assisted grading systems.

7) Training data bias and its impact on grading accuracy: While rubric-driven standardization was applied to enhance consistency in evaluation, the risk of algorithmic bias remains a critical limitation. ChatGPT's training data is predominantly sourced from large public code repositories such as GitHub, which often reflect dominant Western programming idioms and industry-oriented coding styles. These may diverge from academic practices or culturally specific conventions, potentially leading to unfair grading outcomes for students from different pedagogical backgrounds. Such bias may affect both grading fairness and accuracy. Future work could address this issue by fine-tuning the model on institution-specific student code samples and regularly calibrating its evaluations to align with diverse educational standards.

VI. Conclusion

This study evaluated ChatGPT as an automated grading tool for programming assignments across Python, C++, and Java courses, addressing grading accuracy, consistency, rubric adherence, and student perceptions. The findings demonstrate that ChatGPT can serve as a reliable and consistent grading assistant, showing strong alignment with faculty evaluations and high reproducibility across repeated grading attempts. However, its performance is highly dependent on the quality of rubric design and requires human oversight to ensure fairness, mitigate strictness, and handle contextual nuances.

A key contribution of this work is the proposed Instructor–AI Collaborative Rubric Development Framework, which integrates structured rubric design, prompt engineering, and human-in-the-loop validation to support transparent and pedagogically sound assessment. Student perceptions further support the viability of AI-assisted grading, with increased trust and satisfaction following disclosure, although a preference for human grading remains in some cases.

Overall, the results support a hybrid AI–human grading approach that enhances efficiency, consistency, and transparency while maintaining academic integrity. Future research should explore adaptive prompting strategies, fairness auditing across diverse contexts, and the long-term impact of AI-assisted grading on learning outcomes and student behavior.

Declaration on Generative AI Section

During the preparation of this manuscript, the authors used ChatGPT (OpenAI) to improve language and clarity. All outputs were reviewed and edited, and the authors take full responsibility for the final content.

References

[1] H. B. Ajay and A. Others, "Analysis of Essays by Computer (AEC-II). Final Report.," Dec. 1973.

[2] J. R. Carbonell, "AI in CAI: An Artificial-Intelligence Approach to Computer-Assisted Instruction," IEEE Transactions on Man-Machine Systems, vol. 11, no. 4, pp. 190–202, 1970, doi: 10.1109/TMMS.1970.299942.

[3] J. R. Anderson, F. G. Conrad, and A. T. Corbett, "Skill acquisition and the LISP tutor," Cogn Sci, vol. 13, no. 4, pp. 467–505, Oct. 1989, doi: 10.1016/0364-0213(89)90021-9.

[4] V. V. Ramalingam, A. Pandian, P. Chetry, and H. Nigam, "Automated Essay Grading using Machine Learning Algorithm," J Phys Conf Ser, vol. 1000, p. 012030, Apr. 2018, doi: 10.1088/1742-6596/1000/1/012030.

[5] F. Dong, Y. Zhang, and J. Yang, "Attention-based Recurrent Convolutional Neural Network for Automatic Essay Scoring," CoNLL 2017 - 21st Conference on Computational Natural Language Learning, Proceedings, pp. 153–162, 2017, doi: 10.18653/V1/K17-1017.

[6] S. Basu, C. Jacobs, and L. Vanderwende, "Powergrading: a Clustering Approach to Amplify Human Effort for Short Answer Grading," Trans Assoc Comput Linguist, vol. 1, pp. 391–402, Dec. 2013, doi: 10.1162/TACL_A_00236.

[7] S. M. Darwish and S. K. Mohamed, "Automated Essay Evaluation Based on Fusion of Fuzzy Ontology and Latent Semantic Analysis," International Conference on Advanced Machine Learning Technologies and Applications, vol. 921, pp. 566–575, 2019, doi: 10.1007/978-3-030-14118-9_57.

[8] B. Cheang, A. Kurnia, A. Lim, and W. C. Oon, "On automated grading of programming assignments in an academic institution," Comput Educ, vol. 41, no. 2, pp. 121–131, Sep. 2003, doi: 10.1016/S0360-1315(03)00030-7.

[9] M. Messer, N. C. C. Brown, M. Kölling, and M. Shi, "Machine Learning-Based Automated Grading and Feedback Tools for Programming: A Meta-Analysis," Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE, vol. 1, pp. 491–497, Jun. 2023, doi: 10.1145/3587102.3588822.

[10] F. Al Shamsi, A. Elnagar, F. Al Shamsi, and A. Elnagar, "An Intelligent Assessment Tool for Students' Java Submissions in Introductory Programming Courses," Journal of Intelligent Learning Systems and Applications, vol. 4, no. 1, pp. 59–69, Feb. 2012, doi: 10.4236/JILSA.2012.41006.

[11] H. Wang et al., "Examining the applications of intelligent tutoring systems in real educational contexts: A systematic literature review from the social experiment perspective," Educ Inf Technol (Dordr), vol. 28, no. 7, pp. 9113–9148, Jul. 2023, doi: 10.1007/S10639-022-11555-X/TABLES/14.

[12] K. Matthews, T. Janicki, L. He, and L. Patterson, "Implementation of an Automated Grading System with an Adaptive Learning Component to Affect Student Feedback and Response Time," Journal of Information Systems Education, vol. 23, no. 1, pp. 71–83, Jan. 2012. https://aisel.aisnet.org/jise/vol23/iss1/7

[13] C. Geigle, C. Zhai, and D. Ferguson, "An exploration of automated grading of complex assignments," L@S 2016 - Proceedings of the 3rd 2016 ACM Conference on Learning at Scale, pp. 351–360, Apr. 2016, doi: 10.1145/2876034.2876049.

[14] Z. Wang, J. Liu, and R. Dong, "Intelligent Auto-grading System," Proceedings of 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems, CCIS 2018, pp. 430–435, Apr. 2019, doi: 10.1109/CCIS.2018.8691244.

[15] S. Burrows, I. Gurevych, and B. Stein, “The eras and trends of automatic short answer grading,” Int J Artif Intell Educ, vol. 25, no. 1, pp. 60–117, Jan. 2015, doi: 10.1007/S40593-014-0026-8/TABLES/11.

[16] L. Zhang, Y. Huang, X. Yang, S. Yu, and F. Zhuang, “An automatic short-answer grading model for semi-open-ended questions,” Interactive Learning Environments, vol. 30, no. 1, pp. 177–190, 2022, doi: 10.1080/10494820.2019.1648300.

[17] M. Novak and D. Kermek, “Assessment Automation of Complex Student Programming Assignments,” Education Sciences 2024, Vol. 14, Page 54, vol. 14, no. 1, p. 54, Jan. 2024, doi: 10.3390/EDUCSCI14010054.

[18] “Automated Assessment Tool for Teaching Web Application Development (CSEE&T 2024 - Research Track) - CSEE&T 2024.” Accessed: Oct. 17, 2024. [Online]. Available: https://conf.researchr.org/details/cseet-2024/cseet-2024-research-track/38/Automated-Assessment-Tool-for-Teaching-Web-Application-Development

[19] J. K. Matelsky, F. Parodi, T. Liu, R. D. Lange, and K. P. Kording, “A large language model-assisted education tool to provide feedback on open-ended responses,” Jul. 2023, Accessed: Oct. 17, 2024. [Online]. Available: https://arxiv.org/abs/2308.02439v1

[20] R. Montella et al., “Leveraging Large Language Models to Support Authoring Gamified Programming Exercises,” Applied Sciences 2024, Vol. 14, Page 8344, vol. 14, no. 18, p. 8344, Sep. 2024, doi: 10.3390/APP14188344.

[21] M. Messer, N. C. C. Brown, M. Kölling, and M. Shi, “Automated Grading and Feedback Tools for Programming Education: A Systematic Review,” ACM Transactions on Computing Education, vol. 24, no. 1, Feb. 2024, doi: 10.1145/3636515/ASSET/73E610E5-4A8A-40F1-8503-DCA17C688CF5/ASSETS/GRAPHIC/TOCE-2023-0073-F10.JPG.

[22] M. Jukiewicz, “The future of grading programming assignments in education: The role of ChatGPT in automating the assessment and feedback process,” Think Skills Creat, vol. 52, p. 101522, Jun. 2024, doi: 10.1016/J.TSC.2024.101522.

[23] D. Bengtsson and A. Kaliff, “Assessment Accuracy of a Large Language Model on Programming Assignments,” 2023, Accessed: Oct. 17, 2024. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-331000

[24] I. Estévez-Ayres, P. Callejo, M. Á. Hombrados-Herrera, C. Alario-Hoyos, and C. Delgado Kloos, “Evaluation of LLM Tools for Feedback Generation in a Course on Concurrent Programming,” Int J Artif Intell Educ, pp. 1–17, May 2024, doi: 10.1007/S40593-024-00406-0/TABLES/6.

[25] A. Zaid Abualkishik et al., “Outcomes-Based Assessment and Lessons Learned in ABET-CAC Accreditation: A Case Study of the American University in the Emirates,” Mobile Information Systems, vol. 2022, no. 1, p. 1595126, Jan. 2022, doi: 10.1155/2022/1595126.

[26] Cay S. Horstmann and Rance Necaise, Python for Everyone. Wiley, 2020.

[27] “Conover, W.J. (1999) Practical Nonparametric Statistical. 3rd Edition, John Wiley & Sons Inc., New York, 428-433. - References - Scientific Research Publishing.” Accessed: Oct. 20, 2024. [Online]. Available: https://www.scirp.org/reference/referencespapers?referenceid=1496262

[28] T. K. Koo and M. Y. Li, “A Guideline of Selecting and Reporting Intraclass Correlation Coefficients for Reliability Research,” J Chiropr Med, vol. 15, no. 2, p. 155, Jun. 2016, doi: 10.1016/J.JCM.2016.02.012.

[29] OpenAI, “ChatGPT-4,” 2025.

[30] L. W. Anderson, A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom’s Taxonomy of Educational Objectives. New York, NY, USA: Longman, 2001.

[31] S. V. Chinta et al., “FairAIED: Navigating Fairness, Bias, and Ethics in Educational AI Applications,” Jul. 2024.

[32] P. Radanliev, “AI Ethics: Integrating Transparency, Fairness, and Privacy in AI Development,” Applied Artificial Intelligence, vol. 39, no. 1, Dec. 2025, doi: 10.1080/08839514.2025.2463722.

[33] K. Porayska-Pomsta, W. Holmes, and S. Nemorin, “The Ethics of AI in Education,” Mar. 2024, doi: 10.4324/9780429329067.

[34] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou, “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models,” Advances in Neural Information Processing Systems, vol. 35, pp. 24824–24837, 2022.

[35] B. Wang, S. Min, X. Deng, J. Shen, Y. Wu, L. Zettlemoyer, and H. Sun, “Towards Understanding Chain-of-Thought Prompting: An Empirical Study of What Matters,” Proc. 61st Annu. Meeting Assoc. Comput. Linguistics (Vol. 1: Long Papers), Toronto, Canada, 2023, pp. 2717–2739.