Research Journal of Recent Sciences ISSN 2277-2502
Vol. **15(3),** 91-95, July **(2026)** *Res. J. Recent Sci.*

AI-Assisted Assessment of Programming Assignments: A Comparative Study of Automated and Human Evaluation Methods

Asha N.[1*] and Sinchana S.[2]
[1]Department of Computer Science, Nrupathunga University (Formerly Government Science College), Bangalore, Karnataka, India
[2]Department of CSE, BGS College of Engineering and Technology, Bangalore, Karnataka, India
ashan.gsc@gmail.com

Available online at: www.isca.in, www.isca.me
Received 13th April 2026, revised 17th May 2026, accepted 20th June 2026

Abstract

Assessment of programming assignments is an essential part of teaching computer science. Conventional manual grading takes a lot of time and is frequently inconsistent. Using automated code analysis and artificial intelligence methods, the study suggests an Artificial Intelligence (AI) -assisted assessment system for programming assignments. Logical correctness, language syntax, readability, documentation, and code standards are all assessed by the proposed framework. The findings show that while retaining a high degree of agreement with instructor ratings, AI-assisted assessment can drastically cut down on grading time. According to the results, AI can be a useful tool for decision-making to assess a large number of programming assignments to save time.

Keywords: Artificial Intelligence, AI-assessment System, logical correctness, language syntax, code standards.

Introduction

Assignments from students in programming classes must be evaluated on a regular basis. The difficulty of manual assessment increases with class sizes. Large language models and recent developments in artificial intelligence offer chances to automate some aspects of the assessment procedure. This study examines how well AI-assisted assessment systems work and how they might increase scalability, efficiency, and consistency.

Expert code examination, a significant amount of time, and further, in order to evaluate student assignments, it is usually necessary to interact with students for clarification or modifications. The increased availability of improved generative models and code and text processing tools raises the possibility of AI-assisted preliminary grading that enables real-time student work analysis, feedback, and classification. Automated grading systems are becoming more and more required in higher education, particularly with expanding class numbers, to manage high assignment volumes and lighten the workloads of teachers. However, educators acknowledge that AI has the potential to improve assessment procedures, proposing that rather than outlawing such tools, they can be used to increase feedback quality and grading consistency[1].

Results show that the selection of LLM for instructional dissemination is not neutral and could have a significant impact on grading results. The research findings highlight the significance of meticulous model selection and transparent reporting of assessment metrics when incorporating AI-based grading systems in educational settings[2].

By providing immediate, tailored feedback and encouraging an iterative learning process, LLMs improve the learning process. According to the results, LLMs may be crucial to programming education in the future by guaranteeing evaluation consistency and scalability[3].

Enormous automized tools for both static and dynamic computer program evaluation have been published. By reviewing various automated methods for evaluating programming assignments, the author of this paper advances these concerns. Because there are so many tools available, not all of them will be addressed. The article focuses on presenting several methods and strategies for assessment in order to provide an interested reader with a place to start when looking for more information in the field. Teachers can grade assignments with the aid of automatic assessment technologies, and students' work processes can be aided by automatic feedback[4].

This study looks at how well the Large Language Models like ChatGPT-3.5 and GPT-4 perform simple programming jobs Based on the performance, conclusions are drawn about didactic scenarios and LLM-based assessment methods[5].

In order to provide precise and reliable evaluation of programming assignments, our method entails training the LLM on this combined dataset. The goal of the suggested methodology, BeGrading, is to give students prompt, unbiased feedback while lessening the workload associated with grading. Our suggested model shows an absolute difference rate of 19%, or out of 5, when compared to the Codestral model. When utilizing a tiny, refined model with optimum data, this is okay.

Furthermore, there is a 15% discrepancy, or a margin of out of 5, between the dataset optimized score and the Codestral model. Early research indicates that LLMs can perform grading jobs with a high degree of reliability, opening up possibilities for further research and practical applications in automated educational systems[6].

Current developments in natural language processing (NLP), machine learning (ML), and artificial intelligence (AI) have produced a new generation of Large Language Models (LLMs) trained on enormous amounts of data. Thanks to commercial tools like ChatGPT, LLMs may now be used by the general public to produce very excellent-quality texts for scholarly and professional uses. As educational institutions know about how students consume AI-generated content, they are looking into its effects and possible abuse. Because LLMs can produce programming code in multiple languages, computer science (CS) and related industries are especially impacted[7].

In comparison to Individual programming and human-human pair programming methodologies, the study examines how AI-assisted pair programming influences undergraduate Students' performance, programming anxiety, and intrinsic drive[8].

Research Objectives: The following are the research objectives set. i. Assess how well AI-assisted grading works. ii. Contrast teacher evaluations with grades produced by AI.

Objective 1- Assess how well AI-assisted grading works: The aim is to evaluate making use of Artificial Intelligence can reliably and precisely evaluate programming tasks in comparison to human educators. The evaluation criteria are as follows in Table-1.

Table-1: Evaluation Criteria.

Criteria	Description
Logical Correctness	Whether the logic of the program produces the expected output
Syntax	Checking the syntactical grammar of the program
Readability	Meaningful variable names, proper use of identifiers, indentation, and comments.
Coding Standards	adherence to programming standards
Documentation	The calibre of the remarks and justifications

Objective 2- Contrast teacher evaluations with grades produced by AI: A comparative study was made for the grades assigned by human instructors with those generated by an AI-based assessment system for programming assignments. A sample data set of 10 programming assignments is taken to draw the similarities and differences between the human instructor assessment and AI assessment of Programing assignments. The sample data set 0f 10 students for a grade of 40 is as shown in Table-2.

Table-2: Sample Data Set.

Student	Instructor Grade	AI Grade	Difference
S1	35	36	1
S2	22	24	2
S3	38	40	2
S4	17	15	2
S5	15	18	3
S6	28	25	3
S7	31	32	1
S8	36	35	1
S9	10	13	3
S10	21	23	2

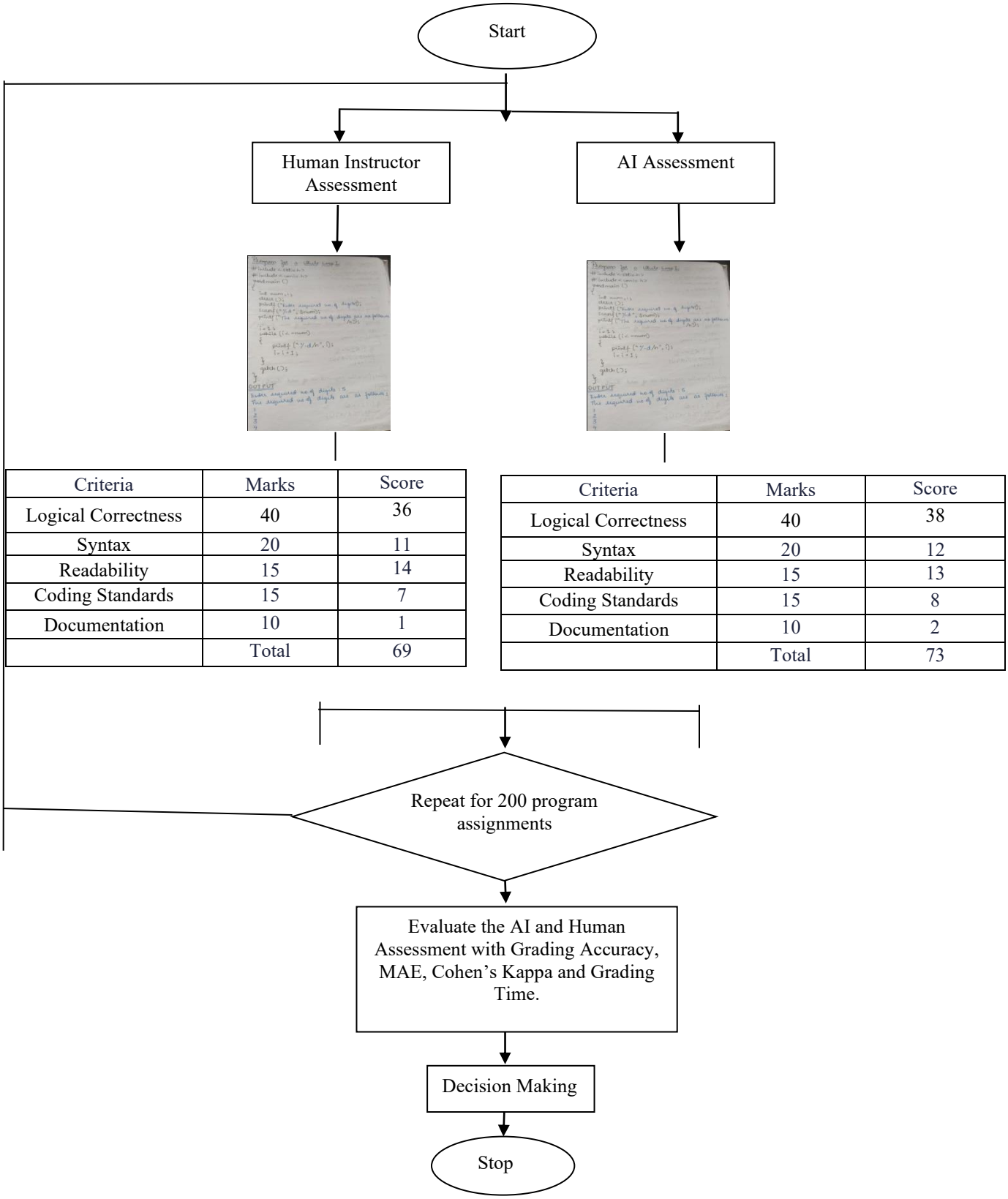
Methodology

Objective-1: The evaluation Criteria to check how well the AI can assess can be justified by collecting 100–200 student programming assignments and comparing the instructor's grade using a standard rubric or rule to evaluate the assignments. The same assignments are used to grade using an AI system like ChatGPT or Gemini[10]. Later, the grading system is compared with human instructor grades and AI grades.

The parameters used to assess the programming assignments are Grading Accuracy, Mean Absolute Error, Cohen's Kappa and Average human instructor Grading Time and Average AI grading Time, is shown in Table-3.

When categorizing items, Cohen's Kappa measures raters' agreement beyond what would be predicted by chance. Despite its widespread use, Cohen's kappa has a number of drawbacks that led to the creation of Gwet's agreement statistic, an alternative "kappa" statistic that uses a "occasional guessing" model to describe chance agreement[9].

The proposed framework is shown in Figure-1 for one assignment which shows the criteria used to assess the programming assignment by the instructor and also AI, which can then be carried out for 200 assignments.



Criteria	Marks	Score
Logical Correctness	40	36
Syntax	20	11
Readability	15	14
Coding Standards	15	7
Documentation	10	1
	Total	69

Criteria	Marks	Score
Logical Correctness	40	38
Syntax	20	12
Readability	15	13
Coding Standards	15	8
Documentation	10	2
	Total	73

Figure -1: Proposed Framework.

Table-3: Parameters assessed for programming assignments by humans and AI[1,3,4].

Parameter	Measurements
Grading Accuracy	Evaluates the degree to which instructor and AI scores are similar. $Accuracy = \frac{\text{Number of Matching Grades}}{\text{Total Grades}} \times 100$
Mean Absolute Error (MAE)	Measures average difference between AI and human scores. $MAE = \frac{1}{n}\sum_{i=1}^{n} \mid A\,I_i - Human_i \mid$
Cohen's Kappa	Measures agreement between AI and human evaluators beyond chance. $K = \frac{P_o - P_e}{1 - P_e}$

Objective 2: Based on the sample dataset (Table-2), the evaluation characteristics between the instructor and AI can be drawn, which is discussed in Table-4.

Table-4: Characteristic Evaluation to assess programming Assignments of Human Instructor and AI.

Characteristics	Human Instructor	AI
Correctness	Very good	Excellent
Creativity	Excellent	Limited
Algorithm Design	Excellent	Good
Readability	Good	Excellent
Time required	High	Very low
Bias Possibility	Moderate	Low

A comparative analysis is also conducted based on the Evaluation Characteristics to highlight the parallels or similarities and discrepancies or differences between the human instructor's and AI's programming assignments. Similarities or the common evaluation criteria adopted by the instructor and the AI is observed and discussed as –Both identify syntax errors. Both assess program correctness or the logic of the program. Both evaluate coding standards.

Table-5: Differences between the instructor and AI.

Feature	Instructor	AI
Grading Speed	Varies from Instructor to Instructor	Greatly Consistent
Creativity Assessment	Strong	Limited
Personalized Feedback	Strong	Moderate
Scalability	Limited	Very High

Results and Discussion

Objective 1: The findings show that AI grading significantly cuts down on grading time while achieving good agreement with instructor judgments. AI excels at evaluating code standards and software accuracy. Nonetheless, human participation is still crucial for assessing outstanding solutions, algorithmic innovation, and creativity.

The study shows that AI-assisted grading can be a useful addition to programming instruction. Although AI cannot fully replace human instructors, it can greatly lessen the workload associated with grading, increase uniformity, and provide pupils with quick feedback. The most realistic option for higher education institutions is a hybrid AI-human assessment architecture. The results of Objective 1 for 200 assignments are as fallows in Table-6.

Table-6: Evaluation of Parameters of Objective 1.

Metric	Result
Grading Accuracy	92.3%
MAE	3.2 Marks
Cohen's Kappa	0.84
Average Human Grading Time	8 min/assignment
Average AI Grading Time	20 sec/assignment

Objective 2: The Statistical Results are estimated shown in Table-7 for similarity and discrepancies discussed in Objective 2. A correlation coefficient of 0.94 indicates a positive relationship between human instructor-assigned grades and AI-generated grades. The low mean absolute error demonstrates that AI scores closely match instructor evaluations.

The comparison shows that AI grading works very well for objective standards like: i. Correctness of the program, ii. Coding guidelines, iii. Records, iv. The capacity to read.

Table-7: Statistical Results.

Metric	Value
Correlation Coefficient	0.94
Mean Absolute Error	2.1 marks
Agreement rate	91%
Time Reduction	85%

But when assessing, Teachers do better than AI: Creative fixes, The inventiveness of algorithms, Methods for solving problems, iv. Design choices that are context-specific.

Conclusion

The results indicate that AI can significantly lessen human instructor burden and consistently help with programming assignment evaluation. However, evaluating higher-order talents like creativity and design thinking still needs human instructors. It is therefore advised to use a mixed AI-human evaluation paradigm, in which instructors evaluate exceptional or borderline entries while AI handles initial grading.

AI-assisted evaluation is a promising method for teaching programming. The suggested Framework reduces the strain for instructors while increasing productivity and uniformity. Future studies can investigate explainable AI methods and adaptive feedback systems for educational evaluation.

References

1. Bernik, A., Radošević, D., & Čep, A. (2025). A comparative study of large language models in programming education: Accuracy, efficiency, and feedback in student assignment grading. *Applied Sciences*, 15(18), 10055.

2. Jukiewicz, M. (2026). A systematic comparison of large language models for automated assignment assessment in programming education: Exploring the importance of architecture and vendor. *Computers and Education Open*, 10, 100364.

3. Mohamed, K., Yousef, M., Medhat, W., Mohamed, E. H., Khoriba, G., & Arafa, T. (2025). Hands-on analysis of using large language models for the auto evaluation of programming assignments. *Information Systems*, 128, 102473.

4. Ala-Mutka, K. M. (2005). A survey of automated assessment approaches for programming assignments. *Computer Science Education*, 15(2), 83–102.

5. Kiesler, N., & Schiffner, D. (2023). Large language models in introductory programming education: ChatGPT's performance and implications for assessments (arXiv Preprint No. arXiv:2308.08572). Cornell University.

6. Yousef, M., Mohamed, K., Medhat, W., Mohamed, E. H., Khoriba, G., & Arafa, T. (2025). BeGrading: large language models for enhanced feedback in programming education. *Neural Computing and Applications*, 37(2), 1027-1040.

7. Raihan, N., Goswami, D., Puspo, S. S. C., Siddiq, M. L., Newman, C., Ranasinghe, T., ... & Zampieri, M. (2026). On the performance of large language models on introductory programming assignments. *Journal of Intelligent Information Systems*, 64(1), 239-263.

8. Fan, G., Liu, D., Zhang, R., & Pan, L. (2025). The impact of AI-assisted pair programming on student motivation, programming anxiety, collaborative learning, and programming performance: A comparative study with traditional pair programming and individual approaches. *International Journal of STEM Education*, 12(1), 16.

9. Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1), 37–46.

10. Zhang, D. W., Boey, M., Tan, Y. Y., & Jia, A. H. S. (2024). Evaluating large language models for criterion-based grading from agreement to consistency. *npj Science of Learning*, 9(1), 79.