

AI-Based Code Assessment in Programming Education: A Systematic Review of Feedback, Reliability, and Ethics

Francisco Genivan Silva¹, Rommel Wladimir de Lima²,
Thiago Reis da Silva³, Mike Christian de Sousa Araujo⁴,
Eduardo Henrique da Silva Aranha¹

¹ Universidade Federal do Rio Grande do Norte (UFRN)

² Universidade do Estado do Rio Grande do Norte (UERN)

³ Instituto Federal de Educação, Ciência e Tecnologia do Maranhão (IFMA)

⁴ Instituto Federal de Educação, Ciência e Tecnologia do Piauí (IFPI)

genivan.silva@ifrn.edu.br, rommelwladimir@uern.br, thiago.reis@ifma.edu.br
mike@ifpi.edu.br, eduardoaranha@dimap.ufrn.br

Abstract. Artificial intelligence (AI) increasingly supports code assessment in programming education, yet evidence of its educational impact, reliability, and responsible use remains fragmented. Following PRISMA guidelines, this systematic review synthesizes 106 empirical studies published from 2021 to 2025. The corpus is concentrated in higher education and introductory programming, with large language models as the most frequent approach, often in hybrid pipelines. Although feedback is common and increasingly explanatory or procedural, only six studies combine experimental or quasi-experimental designs with objective learning indicators. Reliability is stronger for bounded judgements, such as error detection and binary correctness, than for semantic, rubric-based, or code-quality assessment. Reported concerns focus mainly on robustness, generalization, privacy, and compliance, while transparency and governance remain underexplored. Overall, the field is technically promising but pedagogically and methodologically uneven.

1. Introduction

Teaching programming requires frequent feedback on students' code, but providing it at scale remains difficult. Manual assessment supports rich pedagogical judgement, yet it is hard to sustain in large classes or online environments. Automated code assessment has long addressed part of this problem through unit tests, static analysis, and rule-based systems, although these approaches often remain limited to correctness checking or short diagnostic messages.

Recent advances in artificial intelligence (AI) have expanded what automated code assessment can attempt. Machine learning, code representation models, and large language models (LLMs) are now used not only to classify errors or predict grades, but also to generate explanations, hints, and rubric-oriented judgements [Calatayud et al. 2021, Lampou 2023, Al-Hossami et al. 2024, Chen et al. 2023]. This shift matters in programming education because feedback is not only an output of assessment but also a mechanism through which students interpret errors, regulate their learning, and revise their solutions [Hattie and Timperley 2007]. At the same time, AI in assessment introduces

risks related to reliability, transparency, fairness, learner dependence, and academic integrity [Wieser et al. 2023, Porayska-Pomsta et al. 2024, Yan et al. 2024].

AI-based code assessment should therefore not be examined only as a technical problem of prediction or classification. Its educational value depends on how judgements are produced, how feedback is communicated, what evidence supports learning claims, and whether ethical and governance concerns are addressed. Unreliable judgements lead to misleading feedback, weak pedagogical grounding turns fluent explanations into superficial support, and limited ethical reporting obscures the risks of automated decisions in educational settings.

Previous reviews have mapped the evolution of automated programming assessment and the continued relevance of paradigms such as unit tests, static comparisons, and feedback generation [Ala-Mutka 2005, Douce et al. 2005, Messer et al. 2023, Paiva et al. 2022, Keuning et al. 2018]. These syntheses, however, do not jointly examine the instructional quality of AI-generated feedback, the reliability of AI-mediated judgements, the evidence behind learning claims, and the treatment of ethics and governance, although these dimensions are increasingly central to responsible educational AI [Pargman et al. 2024].

Brazilian research has also examined AI in programming education, including classroom applications, automated feedback, and LLM-mediated assessment [Silva et al. 2024, Castilho et al. 2025, Silva and Aranha 2025, Gomes and Hübner 2025]. This work highlights pedagogical opportunities alongside concerns about learner dependence, assessment validity, teacher supervision, and the distinction between apparent performance and actual learning, showing how the international developments reviewed here intersect with local challenges of scalable feedback, pedagogical integration, and responsible adoption.

To address this fragmented evidence base, this study presents a systematic literature review of empirical research on AI-based code assessment in programming education published between 2021 and 2025. The review characterizes the educational contexts in which these systems have been investigated and examines four analytical dimensions, covering technique families and system architectures, feedback and learning evidence, validation of assessment judgements, and barriers related to adoption, ethics, and governance. It is guided by the following research questions:

- RQ1** Which AI technique families and system architectures are most commonly used for automated code assessment?
- RQ2** What types of feedback are provided by AI-based code assessment systems, and what evidence exists regarding their effects on learning?
- RQ3** To what extent do AI-based approaches produce reliable assessment judgements when compared with established oracles such as unit tests, labeled datasets, baseline autograders, or human graders?
- RQ4** What barriers and ethical concerns are reported in the adoption of AI tools for automated code assessment in educational contexts?

The contribution of this review is a critical, cross-dimensional synthesis of a rapidly expanding but uneven field. Across 106 empirical studies, it shows how AI-based code assessment has evolved toward LLM-based and hybrid systems while remaining heterogeneous in learning evidence, validation practices, and ethical operationalization, and

how research stays concentrated in higher education and introductory programming. This concentration is particularly relevant to settings with large classes and limited instructional support, where scalable formative assessment is a persistent challenge. The review therefore identifies conditions that should guide the design of more reliable, pedagogically grounded, and responsible AI-based assessment systems.

2. Methodology

This systematic literature review followed the PRISMA 2020 guidelines [Page et al. 2021] and methodological recommendations for secondary studies in software engineering and computing education [Kitchenham et al. 2015, Kitchenham et al. 2023]. The protocol covered search strategy, eligibility criteria, study selection, quality assessment, data extraction, and synthesis procedures. The review was designed to characterize recent empirical research on AI-based code assessment in programming education and to answer the research questions defined in Section 1.

2.1. Search Strategy

Searches were conducted on October 15, 2025, in Scopus, ACM Digital Library, and IEEE Xplore. These databases were prioritized because the review focuses on programming education and automated code assessment, a topic situated at the intersection of computing education, software engineering education, and educational technology. The database set was defined in the review protocol to prioritize peer-reviewed studies in English available through international bibliographic and publisher databases. This coverage choice may underrepresent Brazilian and Portuguese-language publications not indexed by the selected databases and is therefore treated as a limitation in Section 4.

The corpus was restricted to studies published between January 2021 and October 2025. This temporal scope is a methodological boundary rather than a claim that relevant work began in 2021. Foundational studies and previous reviews are used to contextualize the field, while the primary corpus captures the recent reconfiguration of code assessment under contemporary AI techniques, particularly LLM-based and hybrid systems.

Searches were restricted to title, abstract, and keyword fields to improve precision. The query combined four conceptual blocks connected by AND, with terms within each block connected by OR: AI techniques (e.g., *machine learning*, *LLM*, *generative AI*, *ChatGPT*), assessment processes (e.g., *automated grading*, *evaluation*, *testing*), programming tasks (e.g., *code*, *exercises*, *assignments*), and educational context (e.g., *student*, *teach**, *educat**, *novice*). The final syntax was adapted to each database and benchmarked against previously known relevant studies.

To improve transparency, an anonymized supplementary package accompanies this submission with the database-specific search strings, the list of included studies, the extraction matrices, the quality rubric, and study-level scores.

2.2. Inclusion and Exclusion Criteria

Eligibility criteria were defined to ensure relevance, empirical grounding, and minimum methodological quality. The review included peer-reviewed empirical studies published in journals or conference proceedings, written in English, and published between January

2021 and October 2025. Studies were required to apply or evaluate AI techniques in the context of automated code assessment or feedback for programming education.

The review excluded secondary studies, short papers with fewer than four pages, book chapters, technical reports, non-peer-reviewed publications, and grey literature. During full-text screening, studies were also excluded when they did not evaluate student code, did not describe an AI-based approach with sufficient detail, focused primarily on code generation or problem solving rather than assessment, lacked a clearly defined methodology, or did not report an empirical evaluation. Studies without a clearly reported educational level were retained when their educational context was established through the use of student-code datasets or an explicit focus on assessment or feedback in programming education.

2.3. Selection Process

Study selection was conducted in two stages by a review team composed of four reviewers and one senior adjudicator. First, two reviewers independently screened titles and abstracts, excluding records clearly outside the scope of the review, such as studies from unrelated domains or studies not involving programming education. Disagreements and uncertain cases were resolved through discussion and, when necessary, adjudication by the senior reviewer. Second, potentially eligible full texts were analyzed through cross-reading, with unresolved disagreements decided by the senior adjudicator.

Formal inter-rater agreement indices were not part of the review protocol and were therefore not computed. Instead, consistency was supported by independent dual screening, cross-reading, consensus discussion, and senior adjudication. Consequently, agreement before consensus cannot be quantified, although the reconciliation procedures are explicitly documented. The resulting selection flow is summarized in Figure 1. After full-text screening, 122 studies met the eligibility criteria, and 106 remained in the final corpus after quality assessment.

2.4. Quality Assessment

The methodological quality of the selected studies was evaluated using a checklist adapted from Kitchenham [Kitchenham et al. 2015] and Wohlin [Wohlin et al. 2012]. The checklist comprised ten criteria: clarity of objectives, methodological detail, context description, threats to validity, generalizability, transparency of reporting, support for conclusions, reproducibility, originality, and ethical considerations. Each criterion was scored on a three-point scale, with 1 for fully met, 0.5 for partially met, and 0 for not met.

Each study was scored by one reviewer and audited by a senior reviewer. Disagreements were resolved through discussion until consensus was reached. Studies scoring below six points were excluded from the final synthesis according to a predefined threshold intended to ensure a minimum level of methodological soundness. The complete rubric and study-level scores are included in the anonymized supplementary package.

2.5. Data Extraction and Synthesis

Data extraction followed a structured protocol aligned with the research questions. The extracted data included descriptive metadata, educational context, AI technique family,

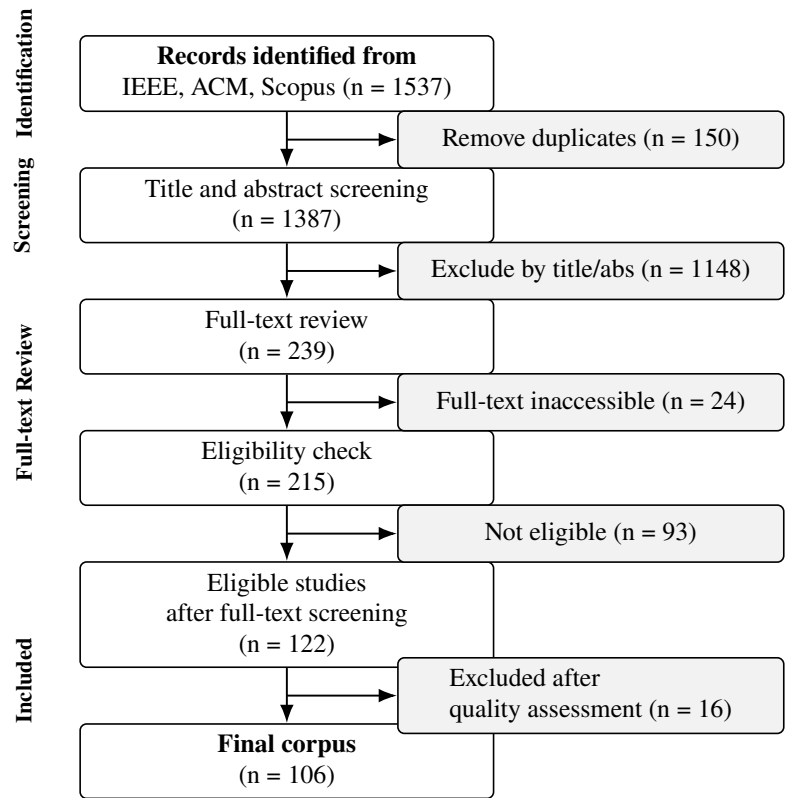


Figure 1. PRISMA flow for study selection.

system architecture, feedback type, evidence of learning effects, assessment target, validation oracle, reported metrics, barriers to adoption, and ethical or governance concerns. Extraction was performed with standardized spreadsheets organized by research question. Entries were audited by a senior reviewer, and ambiguities in the source texts were resolved through discussion.

Because the corpus combined technical studies, educational applications, and empirical evaluations with heterogeneous designs, the synthesis was descriptive and narrative rather than meta-analytic. Frequencies and percentages were used to identify distribution patterns, while cross-tabulations supported comparisons among technique families, feedback types, validation strategies, and reported concerns. Qualitative synthesis was used to interpret recurring patterns and limitations across studies.

The heterogeneity of the corpus is treated as an empirical feature of the field rather than only as a methodological obstacle. For this reason, the analysis does not estimate pooled effects. Instead, it examines how AI-based code assessment has been studied, what types of evidence are used to support claims, and where methodological, pedagogical, and ethical gaps remain.

3. Results

The final corpus comprised 106 empirical studies published between 2021 and 2025. Of the 122 studies that met the eligibility criteria after full-text screening, 16 were excluded after quality assessment (Section 3.1).

The results first report the methodological quality profile of the corpus and char-

acterize the educational contexts in which AI-based code assessment has been investigated. These descriptive analyses provide the background against which the subsequent subsections address the four research questions, focusing on AI techniques and architectures (Section 3.3), feedback and learning evidence (Section 3.4), validation of assessment judgments (Section 3.5), and reported barriers, ethical concerns, and governance issues (Section 3.6).

3.1. Methodological Quality of the Corpus

After applying the predefined quality threshold, 106 studies remained in the final analytical sample. For criterion-level comparison, scores were analyzed on the $[0, 1]$ scale.

The quality assessment indicates an uneven profile across the ten criteria. Reporting-oriented dimensions performed consistently well, including clarity of objectives ($M=0.96$), contextual description ($M=0.95$), transparency of results ($M=0.95$), support for conclusions ($M=0.95$), and originality ($M=0.93$). Methodological adequacy, understood here as the appropriateness of the chosen procedures for the stated objectives, also scored high ($M=0.88$). However, dimensions related to the robustness and transferability of the evidence were considerably weaker, particularly threats to validity ($M=0.51$), generalizability ($M=0.51$), reproducibility ($M=0.63$), and ethical considerations ($M=0.28$).

The mean raw quality score was 7.55 out of 10. This result suggests that most studies provide sufficient information to support descriptive synthesis, but that the evidence base is uneven in important dimensions. Limited reporting on validity, generalization, reproducibility, and ethics constrains the strength of claims that can be drawn from the corpus regarding learning impact, transferability across contexts, and responsible educational deployment.

3.2. Educational Contexts of the Corpus

AI-based code assessment is predominantly reported in higher education. As shown in Table 1, 72 studies (67.9%) are situated in undergraduate contexts, while K–12, postgraduate, and mixed settings together account for fewer than 10% of the corpus. A substantial portion of the studies (25; 23.6%) does not report an identifiable educational level, typically presenting technically oriented approaches evaluated on student-code datasets without describing the instructional environment. Although their educational relevance could be established, this under-specification limits the analysis of pedagogical integration, classroom applicability, and ecological validity.

Within these levels, most studies focus on CS1 or introductory programming courses (57; 53.8%), with smaller shares in broader computing or software engineering courses (13; 12.3%) and interdisciplinary courses involving programming (3; 2.8%). Reported instructional objectives follow the same pattern: 42 studies (39.6%) target introductory programming and logic skills, while only 12 (11.3%) address code quality and style. This concentration should be considered when interpreting the results, since both the technical landscape and the pedagogical claims observed in the corpus are shaped mainly by undergraduate introductory programming.

Table 1. Distribution of studies by educational level (final analysis set, $N=106$).

Educational Level	Number of Studies	% of Total
Undergraduate / Higher Education	72	67.9
Postgraduate	2	1.9
Undergraduate & Postgraduate	2	1.9
High school	1	0.9
Elementary school	3	2.8
K–12 (unspecified)	1	0.9
Not reported / unclear educational context	25	23.6
Total	106	100.0

3.3. RQ1: AI Technique Families and System Architectures

The technical landscape of AI-based code assessment comprises multiple technique families and system architectures. Because many systems combine different methods within the same pipeline, this analysis prioritizes high-level families rather than individual algorithms. A multi-label classification was adopted to capture this architectural overlap and to provide a consistent basis for cross-study comparison.

Table 2 summarizes the predominant technique families and architectural labels. Within the reviewed corpus, LLM-based approaches are the most frequent (62; 58.5%), followed by classical machine learning (28; 26.4%) and deep learning models (21; 19.8%). In addition, 20 studies (18.9%) were identified as hybrid pipelines because they combined at least two technique families. Smaller groups rely on code representation or NLP-for-code models (5; 4.7%) or report techniques generically as neural networks without specifying the underlying architecture (4; 3.8%). Eight studies (7.5%) report nonstandard or insufficiently detailed techniques, while four (3.8%) do not provide an identifiable technique classification or were considered not applicable. Taken together, the categories with limited technical specification are consistent with the reproducibility scores observed in Section 3.1.

Table 2. AI technique families and architectures ($N=106$).

Technique family or architectural label	#Studies	% of N
LLM-based approaches (generative assessment or feedback)	62	58.5
Classical machine learning (engineered features)	28	26.4
Deep learning models (e.g., CNN/RNN/GNN)	21	19.8
Hybrid pipelines (≥ 2 technique families)	20	18.9
Other or insufficiently detailed technique	8	7.5
Code representation / NLP-for-code models (embeddings or encoders)	5	4.7
Neural networks reported generically (ANN)	4	3.8
Technique not specified or not applicable	4	3.8

Note. Percentages use $N=106$ and may exceed 100% because categories overlap.

The predominance of LLM-based approaches should be interpreted with caution. The search strategy was intentionally sensitive to contemporary AI and generative-AI terminology, including terms such as LLM, GPT, and ChatGPT. Therefore, the finding does not imply that LLMs dominate all research on automated code assessment. Rather, it indicates that, within a recent corpus focused on AI-based approaches, LLMs have become a central family since 2023, with non-LLM approaches becoming proportionally

less frequent in subsequent years.

Within the LLM category, several studies report strategies intended to improve reliability, including rubric anchoring and retrieval-augmented generation [Pathak et al. 2025]. Although these are not independent technique families, they suggest movement toward more controlled uses of LLMs in assessment pipelines rather than reliance on prompting alone.

Overall, the corpus suggests that recent AI-based code assessment is increasingly characterized by architectural combinations rather than isolated models, with nearly one in five studies (20; 18.9%) drawing on two or more technique families. At the same time, a share of studies remains technically under-specified, which limits comparability and reproducibility across systems and constrains stronger claims about their transferability to educational settings.

3.4. RQ2: Feedback and Evidence of Learning Effects

This research question separates three issues: whether AI-based code assessment systems provide learner-oriented feedback, what function that feedback serves, and what evidence supports claims about learning effects.

Among the 106 studies, 79 (74.5%) explicitly implement feedback mechanisms, while 27 (25.5%) focus on automated evaluation without pedagogical mediation. Feedback-providing studies were grouped into three functional categories informed by feedback theory, particularly Hattie and Timperley [Hattie and Timperley 2007]. Corrective feedback indicates an error or correctness state with little or no explanation; explanatory feedback clarifies why a response is correct or incorrect or explains the underlying issue; and procedural feedback offers hints, steps, or strategies to guide revision. The resulting distribution was 16 studies with corrective feedback (20.3% of the feedback subset), 23 with explanatory feedback (29.1%), and 40 with procedural feedback (50.6%). These categories describe the intended function of the feedback rather than evidence of its educational effectiveness.

The prevalence of explanatory and procedural feedback indicates a movement toward more instructive forms of mediation, aligned with the diffusion of LLM-based systems discussed in Section 3.3. However, procedural feedback was often described only in generic terms such as “hints” or “step-by-step guidance”, suggesting that elaborate feedback frequently appears as a technical capability of generative models rather than as the result of deliberate pedagogical design. Moreover, the corpus rarely shows whether learners actually use such guidance to revise their approach.

Learning evidence was analyzed separately according to study design, measurement source, and reported outcome. Only 16 studies (15.1% of the corpus) report any measurement aimed at learning. Among them, three use experimental designs with control groups [Hou et al. 2024, Naik et al. 2024, Alrabah and Alawini 2025], five use quasi-experimental designs with intra-group or historical comparisons [Kaleem et al. 2024, Troussas et al. 2023, Cámara et al. 2024, Roldán-Álvarez and Mesa 2024, Qi et al. 2025], and eight are classroom experience reports [Taeb and Chi 2021, Frankford et al. 2024, Sanchez-Anguix et al. 2023, Vadaparty et al. 2024, Denny et al. 2024a, Kimmel et al. 2024, Denny et al. 2024b, Berrezueta-Guzman and Krusche 2023]. When measurement design and evidence source

are considered together, only 6 studies (5.7% of the corpus, or 37.5% of those attempting learning measurement) combine an experimental or quasi-experimental design with objective or quasi-objective indicators such as test scores or course grades. This pattern is consistent with the weaker validity, generalizability, and reproducibility scores reported in Section 3.1.

Across these 16 studies, outcomes vary. Five studies with experimental or quasi-experimental designs using objective indicators or course grades report positive effects [Kaleem et al. 2024, Troussas et al. 2023, Hou et al. 2024, Cámara et al. 2024, Roldán-Álvarez and Mesa 2024]. Three studies report mixed, null, or potentially adverse effects, particularly in introductory programming contexts where LLM-based assistance may coincide with weaker independent performance [Vadaparty et al. 2024, Kimmel et al. 2024, Naik et al. 2024]. The remaining eight studies rely on perception-based or inconclusive evidence, including experience reports and quasi-experiments using usage proxies, for example [Frankford et al. 2024, Denny et al. 2024a]. Although such studies often report positive attitudes, they do not establish actual learning improvement. The full classification by design, measurement base, and reported outcome is provided in the supplementary package.

Taken together, the corpus suggests that AI-based feedback can support learning under specific conditions, but current evidence does not justify broad claims of learning effectiveness. The main gap is not the absence of feedback mechanisms, but the limited availability of robust empirical designs connecting feedback use to learning outcomes, revision behavior, or longer-term development of programming competence.

3.5. RQ3: Reliability of AI-Based Assessment Judgments

This research question examines what current studies allow us to conclude about the reliability of AI-based assessment judgments. While RQ2 focused on whether feedback supports learning, this section addresses an underlying condition for any pedagogical use: whether the judgments produced by AI systems are consistent with reference standards. An assessment judgment, in this analysis, refers to any output used to evaluate student code, including pass or fail decisions, numeric scores, rubric ratings, error categories, semantic adequacy, or code-quality judgments.

Of the 106 studies in the corpus, 94 (88.7%) explicitly evaluate assessment outputs. The remaining 12 (11.3%) propose supporting components such as recommendation frameworks, feedback generators, or auxiliary tools, without treating assessment judgments as the primary object of validation. The analysis below focuses on these 94 studies and shows that the strength of reliability evidence depends on the interaction among the type of judgment being evaluated, the oracle used as reference, and the breadth of the validation setting.

The judgments with stronger validation evidence are those anchored in observable outcomes. Within the assessment-focused subset, studies frequently evaluate error detection (24; 25.5%), numeric scoring (23; 24.5%), and binary correctness (22; 23.4%). When these targets are paired with test-based oracles such as unit tests or online judges (14; 14.9%) or with curated labeled datasets (23; 24.5%), validation tends to be more transparent and metrics easier to interpret. Higher-order judgments, including rubric-based ratings (9; 9.6%), semantic understanding (6; 6.4%), and explicit code-quality evalua-

tion (2; 2.1%), are less frequently studied and harder to validate because their reference standards involve more interpretive judgment. A further 8 studies (8.5%) evaluate other assessment-related targets, such as collaboration assessment, design pattern detection, or emotional state estimation.

Reliability claims become more difficult to assess when the oracle itself is subjective. The largest group of studies compares AI judgments against human graders (40; 42.6%), which establishes agreement with grading practice but does not provide an independent functional oracle unless the grading process is externally validated. Mixed-oracle strategies that combine tests with human review remain rare (7; 7.4%), and explicit comparisons with baseline autograders appear in only four studies (4.3%). The result is that a substantial portion of the validation literature shows agreement with existing grading judgments rather than convergence across independent sources of evidence.

A further limitation concerns the breadth of validation. Of the 94 assessment-focused studies, 80 (85.1%) report quantitative validation using measurable indicators such as classification metrics, agreement measures, or correlation measures. Although these indicators are appropriate for the targets being evaluated, the validation settings are often narrow, such as offline datasets, single-task configurations, or homogeneous exercises. Few studies report behavior under distribution shifts such as unseen problems, alternative solution strategies, multiple programming languages, or new learner cohorts. Strong metric values within a controlled setting therefore offer limited assurance about deployment in classroom conditions.

Taken together, these patterns suggest a graduation of reliability across the corpus. AI-based assessment is most defensible when the judgment is bounded, the oracle is verifiable, and the validation includes diverse conditions, and progressively less defensible as any of these conditions weakens. This graduation matters pedagogically: feedback generated from less reliable judgments inherits that uncertainty, even when the feedback text appears elaborate or contextualized.

3.6. RQ4: Barriers, Ethical Concerns, and Governance Issues

This research question examines the barriers and ethical concerns reported in the corpus, identifying conditions that may constrain the educational adoption of AI-based code assessment.

The most frequently reported barrier concerns robustness and reliability (50; 47.2%). Recurrent issues include false positives and false negatives, inaccurate guidance, instability across repeated runs, hallucinations, and overly generic feedback, especially in LLM-based systems [Hellas et al. 2023, Frankford et al. 2024, Lagakis and Demetriadis 2024, Al-Hossami et al. 2024]. These findings reinforce the analysis in RQ3, since reliability concerns are not limited to aggregate metrics but appear as concrete failure modes that affect trust and pedagogical usefulness.

The second most frequent barrier concerns data dependency and limited generalization (32; 30.2%). Studies report difficulty handling diverse student solution strategies, task formats outside the training distribution, underspecified assignments, multi-file programs, and heterogeneous course materials [Li et al. 2024, Sokač et al. 2025, Roldán-Álvarez and Mesa 2024]. Strong results in constrained settings therefore do not necessarily imply robust behavior across new problems, cohorts, or instructional contexts.

Less frequent barriers cover qualitatively distinct concerns, including transparency and explainability (6; 5.7%) [Dong and Liang 2024, Rico-Juan et al. 2023], pedagogical risks and misuse (5; 4.7%) [Vadaparty et al. 2024, Berrezueta-Guzman and Krusche 2023], and infrastructure or usability constraints (6; 5.7%). Although less common than robustness and generalization, these barriers are directly relevant when AI-based assessment moves from optional support toward consequential roles.

Ethical reporting in the corpus is uneven and often shallow. Nearly half of the studies (51; 48.1%) provide no explicit ethical discussion. Among the 55 (51.9%) that do, the discussion concentrates on privacy, anonymization, consent, or institutional approval. Broader concerns associated with responsible AI in education appear far less frequently, including academic integrity and misuse (16; 15.1%), fairness and bias (10; 9.4%), transparency and accountability (5; 4.7%), learner dependency (5; 4.7%), and security (3; 2.8%). The depth is also limited: most references are procedural (29; 52.7%) or brief (20; 36.4%), and only six (10.9%) provide deeper analytical discussion. This pattern is consistent with the methodological quality profile in Section 3.1, in which ethical considerations received the lowest mean score across all ten criteria. When compared with established frameworks for responsible AI in education [Miao and Holmes 2023], several of these dimensions, particularly those involving contestability of automated decisions and learner protection, remain marginal in the empirical literature.

Taken together, the barriers reported in the corpus reinforce the central pattern of this review. AI-based code assessment is advancing rapidly in technical capability, but its educational adoption remains constrained by reliability, generalization, transparency, and limited ethical operationalization. These constraints are especially relevant when systems move from low-stakes feedback support toward decisions that may influence assessment, progression, or learner behavior.

4. Discussion and Threats to Validity

The overall picture is a field whose technical capability has grown faster than its pedagogical and methodological maturity. LLM-based and hybrid systems dominate recent research, yet the strength of the available evidence depends on what is being judged, how judgments are validated, and how responsibly systems are reported. This discussion connects these dimensions, since they interact in ways that the individual results do not fully reveal.

Reliability emerges from the corpus as a conditional property rather than an attribute of any technique. Validation is most convincing when the judgment is bounded, as in error detection, binary correctness, and numeric scoring, and when the reference standard is verifiable. The oracle also delimits what a metric can claim. Unit tests indicate functional adequacy, human graders reflect rubric-based practice, and course grades combine multiple curricular influences, so agreement with one reference does not transfer to the others. Higher-order judgments involving semantics, rubrics, and code quality remain less studied and depend on more interpretive standards. Studies should therefore state what is validated, against which reference, and under what transfer conditions. This need is most visible in the 25 studies without a reported educational level, whose dataset-based evidence, although educationally relevant, offers limited support for claims about

classroom integration and ecological validity.

These reliability conditions propagate directly to feedback. Explanatory or procedural outputs may appear pedagogically convincing even when the underlying diagnosis is unstable, and many systems present hints or step-by-step guidance as generative capabilities rather than as choices grounded in feedback theory [Hattie and Timperley 2007]. Because only six studies combine experimental or quasi-experimental designs with objective learning indicators, most learning claims rest on perceptions or immediate task performance. Such evidence may create an illusion of competence, in which fluent assistance is mistaken for durable understanding or independent programming ability, so learning claims should remain conditional on instructional integration, measurement design, and assessment validity.

A similar distance separates ethical awareness from ethical operationalization. Reporting concentrates on privacy, consent, anonymization, and institutional approval, while transparency, fairness, accountability, learner dependence, and contestability remain marginal despite existing guidance for responsible educational AI [Miao and Holmes 2023, Castilho et al. 2025]. These dimensions are also relevant to Brazilian contexts seeking scalable feedback without weakening teacher mediation, and they become decisive as systems move toward consequential assessment, which requires auditable judgments and clear mechanisms for human review.

The review itself has coverage and classification limitations. Searches restricted to Scopus, ACM Digital Library, and IEEE Xplore may have missed studies indexed mainly in educational databases, as well as Brazilian, Portuguese-language, grey-literature, and unavailable publications. Interpretive categories are not standardized across the literature, and no formal inter-rater agreement index was computed before consensus. The primary evidence is also heterogeneous, and publication bias may underrepresent failures. The findings therefore describe dominant patterns and gaps rather than a definitive estimate of effectiveness.

5. Concluding remarks and future directions

This review shows that recent AI-based code assessment is increasingly organized around controlled uses of LLMs embedded in hybrid pipelines, rather than around the simple replacement of earlier paradigms by generative models. At the same time, technical capability has advanced faster than pedagogical validation and ethical operationalization. Research remains concentrated in introductory programming and higher education, validation privileges bounded targets over higher-order judgments, and evidence of learning effects is limited and methodologically uneven.

Addressing these gaps will require empirical work combining experimental or quasi-experimental designs with objective learning indicators, validation strategies that report behavior across cohorts and task variations, and stronger integration of ethical and governance concerns beyond privacy and compliance. AI-based code assessment is best understood as a socio-technical educational system whose value depends on the alignment between technical design, instructional integration, and robust evaluation. Beyond programming education, the review contributes to broader debates on responsible AI in learning environments by clarifying the conditions under which automated assessment may become more trustworthy, pedagogically meaningful, and ethically grounded.

Artifact Availability

The supplementary artifacts of this study are available at <https://github.com/genivansilva/sbie2026-slr-ia-artifacts.git>.

The package includes the database-specific search strings, the list of included studies, the educational-context file, the research-question extraction files, the quality assessment criteria and study-level scores, and a README describing the files.

AI Use Statement

The authors used ChatGPT as a support tool during data analysis and manuscript preparation. The researchers performed the primary data extraction and produced the structured spreadsheets used in the review. These spreadsheets were subsequently processed with ChatGPT to support data organization and tabulation, the computation and verification of descriptive metrics, and the generation of preliminary summaries and analytical insights. ChatGPT was also used for language editing. It was not used to conduct database searches, determine study eligibility, or make final methodological and interpretive decisions. All AI-assisted outputs were reviewed and validated by the authors, who assume full responsibility for the scientific content and conclusions of the article.

References

- Al-Hossami, E., Bunescu, R., Smith, J., and Teehan, R. (2024). Can language models employ the socratic method? experiments with code debugging. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2024)*, pages 53–59. Association for Computing Machinery.
- Ala-Mutka, K. M. (2005). A survey of automated assessment approaches for programming assignments. *Computer Science Education*, 15(2):83–102.
- Arabab, A. and Alawini, A. (2025). Codelens: A generative ai framework for automated feedback on sql assignments. In *Proceedings of the 4th International Workshop on Data Systems Education: Bridging Education Practice with Education Research*, DataEd '25, page 29–34, New York, NY, USA. Association for Computing Machinery.
- Berrezueta-Guzman, J. and Krusche, S. (2023). Recommendations to create programming exercises to overcome chatgpt. In *2023 IEEE 35th International Conference on Software Engineering Education and Training (CSEE&T)*, pages 147–151.
- Calatayud, V. G., Espinosa, M. P. P., and Vila, R. R. (2021). Artificial intelligence for student assessment: A systematic review. *Applied Sciences*, 11(12):5467.
- Castilho, G. U., Herrera, V. A., and Rodriguez, C. L. (2025). Explorando o potencial da ia generativa em ambientes virtuais de aprendizagem: Uma revisão sistemática. In *Anais do XXXVI Simpósio Brasileiro de Informática na Educação*, SBIE 2025, pages 113–126. Sociedade Brasileira de Computação.
- Chen, E., Huang, R., Chen, H.-S., Tseng, Y.-H., and Li, L.-Y. (2023). Gptutor: A chatgpt-powered programming tool for code explanation. In Wang, N., Rebolledo-Mendez, G., Dimitrova, V., Matsuda, N., and Santos, O., editors, *Artificial Intelligence in Education. Posters and Late Breaking Results, Workshops and Tutorials, Industry and Innovation Tracks, Practitioners, Doctoral Consortium and Blue Sky*, volume 1831

- of *Communications in Computer and Information Science*, pages 632–647. Springer, Cham.
- Cámara, J., Troya, J., Montes-Torres, J., and Jaime, F. J. (2024). Generative ai in the software modeling classroom: An experience report with chatgpt and unified modeling language. *IEEE Software*, 41(06):73–81.
- Denny, P., MacNeil, S., Savelka, J., Porter, L., and Luxton-Reilly, A. (2024a). Desirable characteristics for ai teaching assistants in programming education. In *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*, ITiCSE 2024, page 408–414, New York, NY, USA. Association for Computing Machinery.
- Denny, P., Smith IV, D. H., Fowler, M., Prather, J., Becker, B. A., and Leinonen, J. (2024b). Explaining code with a purpose: An integrated approach for developing code comprehension and prompting skills. In *Proceedings of the 2024 Conference on Innovation and Technology in Computer Science Education (ITiCSE)*.
- Dong, D. and Liang, Y. (2024). Grading programming assignments by summarization. In *Proceedings of the ACM Turing Award Celebration Conference - China 2024*, ACM-TURC '24, page 53–58, New York, NY, USA. Association for Computing Machinery.
- Douce, C., Livingstone, D., and Orwell, J. (2005). Automatic test-based assessment of programming: a review. *Journal on Educational Resources in Computing (JERIC)*, 5(3):Article 4–es.
- Frankford, E., Sauerwein, C., Bassner, P., Krusche, S., and Breu, R. (2024). Ai-tutoring in software engineering education. In *Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training*, ICSE-SEET '24, page 309–319, New York, NY, USA. Association for Computing Machinery.
- Gomes, P. C. R. and Hübner, J. F. (2025). Avaliação da aprendizagem em programação com IA generativa na educação profissional e tecnológica. In *Anais do XXXVI Simpósio Brasileiro de Informática na Educação*, pages 1529–1539, Porto Alegre, RS, Brasil. Sociedade Brasileira de Computação.
- Hattie, J. and Timperley, H. (2007). The power of feedback. *Review of Educational Research*, 77(1):81–112.
- Hellas, A., Leinonen, J., Sarsa, S., Koutchene, C., Kujanpää, L., and Sorva, J. (2023). Exploring the responses of large language models to beginner programmers' help requests. In *Proceedings of the 2023 ACM Conference on International Computing Education Research (ICER '23)*, volume 1, pages 93–105. Association for Computing Machinery.
- Hou, X., Wu, Z., Wang, X., and Ericson, B. J. (2024). Codetailor: Llm-powered personalized parsons puzzles for engaging support while learning programming. In *Proceedings of the Eleventh ACM Conference on Learning @ Scale (L@S '24)*, pages 51–62. Association for Computing Machinery.
- Kaleem, M., Hassan, M. A., Anwar, Z., and Ehsan, N. (2024). A machine learning-based adaptive feedback system to enhance programming skill using computational thinking. In *2024 IEEE Global Engineering Education Conference (EDUCON)*. IEEE.

- Keuning, H., Jeurig, J., and Heeren, B. (2018). A systematic literature review of automated feedback generation for programming exercises. *ACM Transactions on Computing Education*, 19(1):1–43.
- Kimmel, B., Geisert, A. L., Yaro, L., Gipson, B., Hotchkiss, R. T., Osae-Asante, S. K., Vaught, H., Wininger, G., and Yamaguchi, C. (2024). Enhancing programming error messages in real time with generative ai. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, CHI EA '24, New York, NY, USA. Association for Computing Machinery.
- Kitchenham, B., Budgen, D., and Brereton, P. (2015). *Evidence-Based Software Engineering and Systematic Reviews*. Chapman and Hall/CRC.
- Kitchenham, B., Madeyski, L., and Budgen, D. (2023). Segress: Software engineering guidelines for reporting secondary studies. *IEEE Transactions on Software Engineering*, 49(3):1273–1298.
- Lagakis, P. and Demetriadis, S. (2024). Evaai: A multi-agent framework leveraging large language models for enhanced automated grading. In Sifaleras, A. and Lin, F., editors, *Generative Intelligence and Intelligent Tutoring Systems. ITS 2024. Lecture Notes in Computer Science*, volume 14798, page 378–385. Springer, Cham.
- Lampou, R. (2023). The integration of artificial intelligence in education: Opportunities and challenges. *Review of Artificial Intelligence in Education*, 4(00):e015.
- Li, J., Zhao, Y., Li, Y., Li, G., and Jin, Z. (2024). Acecoder: An effective prompting technique specialized in code generation. *ACM Trans. Softw. Eng. Methodol.*, 33(8).
- Messer, M., Brown, N. C. C., Kölling, M., and Shi, M. (2023). Machine learning based automated grading and feedback tools for programming: A meta analysis. In *Proceedings of ITiCSE 2023*, pages 491–497. ACM.
- Miao, F. and Holmes, W. (2023). Guidance for generative AI in education and research. Technical report, UNESCO, Paris, France.
- Naik, A., Yin, J. R., Kamath, A., Ma, Q., Wu, S. T., Murray, C., Bogart, C., Sakr, M., and Rose, C. P. (2024). Generating situated reflection triggers about alternative solution paths: A case study of generative ai for computer-supported collaborative learning. In *Artificial Intelligence in Education: 25th International Conference, AIED 2024, Recife, Brazil, July 8–12, 2024, Proceedings, Part I*, page 46–59. Springer-Verlag, Berlin, Heidelberg.
- Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., McGuinness, L. A., Stewart, L. A., Thomas, J., Tricco, A. C., Welch, V. A., Whiting, P., and Moher, D. (2021). The prisma 2020 statement: an updated guideline for reporting systematic reviews. *BMJ*, 372:n71.
- Paiva, J. C., Leal, J. P., and Figueira, A. (2022). Automated assessment in computer science education: A state-of-the-art review. *ACM Trans. Comput. Educ.*, 22(3).

- Pargman, T. C., McGrath, C., and Milrad, M. (2024). Towards responsible ai in education: Challenges and implications for research and practice. *Computers and Education: Artificial Intelligence*, page 100345.
- Pathak, A., Gandhi, R., Uttam, V., Ramamoorthy, A., Ghosh, P., Jindal, A. R., Verma, S., Mittal, A., Ased, A., Khatri, C., Nakka, Y., Devansh, Challa, J. S., and Kumar, D. (2025). Rubric is all you need: Improving llm-based code evaluation with question-specific rubrics. In *Proceedings of the 2025 ACM Conference on International Computing Education Research V.1, ICER '25*, page 181–195, New York, NY, USA. Association for Computing Machinery.
- Porayska-Pomsta, K., Holmes, W., and Nemorin, S. (2024). The ethics of ai in education. In du Boulay, B., Mitrovic, A., and Yacef, K., editors, *Handbook of Artificial Intelligence in Education*, pages 571–604. Edward Elgar.
- Qi, L., Zamfirescu-Pereira, J., Kim, T., Hartmann, B., DeNero, J., and Norouzi, N. (2025). A knowledge-component-based methodology for evaluating ai assistants. In *Proceedings of the ACM Global on Computing Education Conference 2025 Vol 1, CompEd 2025*, page 78–84, New York, NY, USA. Association for Computing Machinery.
- Rico-Juan, J. R., Sánchez-Cartagena, V. M., Valero-Mas, J. J., and Gallego, A. J. (2023). Identifying student profiles within online judge systems using explainable artificial intelligence. *IEEE Transactions on Learning Technologies*, 16(6):955–969.
- Roldán-Álvarez, D. and Mesa, F. J. (2024). Intelligent deep-learning tutoring system to assist instructors in programming courses. *IEEE Transactions on Education*, 67(1):153–161.
- Sanchez-Anguix, V., Alberola, J. M., Del Val, E., Palomares, A., and Teruel, M. D. (2023). Comparing computational algorithms for team formation in the classroom: a classroom experience. *Applied Intelligence*, 53(20):23883–23904.
- Silva, F. G. and Aranha, E. H. S. (2025). Feedback formativo automatizado com LLMs: Desenvolvimento e análise de um sistema para aprendizagem progressiva em programação. In *Anais do XXXVI Simpósio Brasileiro de Informática na Educação*, pages 1159–1173, Curitiba, PR, Brasil.
- Silva, T. L. d., Vidotto, K. N. S., Tarouco, L. M. R., and Silva, P. F. d. (2024). Potencialidades do uso de inteligência artificial generativa como apoio ao ensino de programação. In *Anais do XXXV Simpósio Brasileiro de Informática na Educação*, pages 1942–1956, Porto Alegre, RS, Brasil. Sociedade Brasileira de Computação.
- Sokač, M., Fabijanić, M., Mekterović, I., and Mršić, L. (2025). Automated grading through contrastive learning: A gradient analysis and feature ablation approach. *Machine Learning and Knowledge Extraction*, 7(2):41.
- Taeb, M. and Chi, H. (2021). A personalized learning framework for software vulnerability detection and education. In *2021 International Symposium on Computer Science and Intelligent Controls (ISCSIC)*, pages 119–126.
- Troussas, C., Krouska, A., and Virvou, M. (2023). A multilayer inference engine for individualized tutoring model: adapting learning material and its granularity. *Neural Comput & Applic*, 35:61–75.

- Vadaparty, A., Zingaro, D., Smith IV, D. H., Padala, M., Alvarado, C., Gorson Benario, J., and Porter, L. (2024). Cs1-llm: Integrating llms into cs1 instruction. In *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*, ITiCSE 2024, pages 297–303, New York, NY, USA. Association for Computing Machinery.
- Wieser, M., Röder, F., Helmer, S., and Weitz, J. (2023). Investigating the role of chatgpt in supporting text-based programming education for students and teachers. *Computers and Education: Artificial Intelligence*, 5:100232.
- Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in Software Engineering*. Springer.
- Yan, L., Sha, L., Zhao, L., Li, Y., Maldonado, R. M., Chen, G., Li, X., Jin, Y., and Gašević, D. (2024). Practical and ethical challenges of large language models in education: A systematic scoping review. *British Journal of Educational Technology*, 55(1):90–112.