JPPIPA 11(12) (2025)

Jurnal Penelitian Pendidikan IPA

http://jppipa.unram.ac.id/index.php/jppipa/index

Student Flowchart Automated Evaluation for Scalable Assessment in Introductory Programming

Usman Nurhasan[1,2], Didik Dwi Prasetya[1*]

[1] Department of Electrical Engineering and Informatics, Faculty of Engineering, Universitas Negeri Malang, Indonesia
[2] Department of Information Technology, Politeknik Negeri Malang, Indonesia

Received: October 25, 2025
Revised: November 30, 2025
Accepted: December 25, 2025
Published: December 31, 2025

Corresponding Author:
Didik Dwi Prasetya
didikdwi@um.ac.id

DOI: 10.29303/jppipa.v11i12.13594

Abstract: This study evaluates the Automated Flowchart Assessment Tool (AFAT) to overcome limitations in semantic sensitivity and layout robustness prevalent in existing tools. Through a quantitative analysis of 312 student submissions, AFAT demonstrated superior diagnostic performance with a Micro-F1 score of 0.92 and substantial inter-rater agreement (Fleiss' Kappa = 0.88), supporting the hypothesis of expert-level accuracy. Key findings reveal that AFAT significantly enhances operational efficiency, reducing evaluation time by 61.2% (averaging 1.87 minutes per flowchart) while decreasing inter-rater variability by 28%. Generalized Linear Model (GLM) analysis confirmed significant time savings, particularly in high-complexity sessions (Wald $\chi^2 = 87.44$, $p < 0.001$). Beyond technical efficiency, this research contributes to applied science education by providing a scalable framework for computational science literacy, enabling the rigorous assessment of algorithmic thinking within integrated STEM curricula. These results substantiate AFAT's potential for large-scale deployment as a robust tool for automated scoring in formal educational settings.

Keywords: Diagnostic Accuracy; Evaluation Efficiency; Scoring Reliability; Flowchart Assessment; Semantic Robustness

Introduction

The rapid global integration of programming into K-12 curricula has triggered a scalability crisis in formative assessment, where instructors struggle to provide timely, individualized, and semantically accurate feedback as student numbers rise (Florou et al., 2024). This delay in evaluating visual artifacts like flowcharts significantly hinders students' reflective learning and algorithmic thinking (Sakulin et al., 2025). Unlike syntax-heavy source code that facilitates automated output-based testing, flowchart assessment demands a complex interpretation of semantic, spatial, and topological structures (Calderon et al., 2023). Consequently, conventional rule-based systems often fail to recognize functionally equivalent logic presented in diverse visual layouts, while current NLP and heuristic approaches remain limited by low semantic accuracy and excessive reliance on manual transcription (Messer et al., 2024).

In East Java's vocational higher education contexts, assessment of flowchart exercises in introductory programming courses is carried out manual by a group of four instructors who manage seven parallel classes of 30 students each. During a 16-week semester, each teaching session begins with the construction of flowcharts, which serves as a precursor to coding. Data gathered from 18 instructors show an average correction workload of 5.8 hours a week, inter-rater inconsistency of 26%, and variability of evaluative interpretation to the degree of significance(Tong et al., 2023; Weegar & Idestam-almquist, 2023). Although these findings are context-specific, the assessments burdens placed on instructors on CS education manual assessments suggest pressing need for scalability.

Globally, flowchart-based instruction is the taught foundational programming structure and remains the curricula for K-12 and vocational education(Lee et al., 2023)(Ye et al., 2023). The assessment of flowcharts stands as a bottleneck in the challenges posed by the

How to Cite:
Nurhasan, U., & Prasetya, D. D. (2025). Student Flowchart Automated Evaluation for Scalable Assessment in Introductory Programming. *Jurnal Penelitian Pendidikan IPA*, *11*(12), 1230–1240. https://doi.org/10.29303/jppipa.v11i12.13594

educational system and the logic of visual programming(C. Huang et al., 2025). However, for research relevance, flowchart grading will need to be enrolled as prerequisite course to the scalable assessment of advanced visual programming elements such as UML, State machines and algorithmic maps (Zimmerman et al., 2023).

In order to mitigate the problems discussed above, this research constructs an automated flowchart evaluation system based on a teacher-model comparison. While many constructed flowchart systems utilize Control Flow Graphs (CFGs) and Program Dependency Graphs (PDGs) which are designed with a compiler's perspective, this research takes a pedagogical perspective(Xu et al., 2025). By contrast, Directed Graphs (DG) permit a flexible structure for conceptualizing student logic without block-level semantics, thus allowing layout-tolerant comparisons. In this system, the proposed new topological distance metric is introduced and the formal definition is within the methodology. This new metric is designed to quantify the degree of semantic path equivalence of student and teacher graphs(Chen et al., 2019). Graph Edit Distance (GED) and Maximum Common Subgraph (MCS) are traditional metrics, yet they are not practical because of the exponential cost associated with cyclic structures(Dikici & Bilgin, 2025). Unlike those metrics, this one abstract control flow structures of loops, branches, and sequences into a weighted, pedagogically relevant path. The rest of the feature engineering follows the classical constructs of Computer Science education, namely, AST-based abstraction combined with control flow pattern extraction (Geetika et al., 2025).

For empirical rigor, this system uses the automated grading tool GRAD-AI, which is an advanced system for grading visual logic and code artifacts (Gambo et al., 2024). This automated system, coupled with expert grading, provides a reproducible benchmark for assessing the system's classification accuracy. The efficacy of the system is analyzed through the following two technical hypotheses.

The AFAT system is designed to achieve a logical classification accuracy 90% (H1) and a 60% reduction in evaluation time (H2), ensuring high inter-rater reliability through separate calibration and validation phases. To enhance generalizability, the framework incorporates UML and state machine extensions, utilizing advanced topological metrics to interpret hierarchical and concurrent transitions (Cui et al., 2024). This research addresses a significant gap in existing literature: the absence of scalable, semantically accurate tools for evaluating visual logic in interdisciplinary contexts. The novelty of AFAT lies in its integration of graph theory and control flow analysis with educational assessment to foster computational science literacy. By delivering high-fidelity automated feedback, the system directly enriches science education, empowering students to model complex phenomena while systematically sharpening their algorithmic thinking skills.

Method

Context and Participants

This study took place in one of the vocational higher education institutions in East Java, Indonesia, during the introductory fundamental programming course. This course is offered in the first semester and lasts for 16 weeks, comprising 48 contact hours. The course syllabus covers elementary content areas such as algorithmic thinking, control structures, and flowchart construction, and is geared toward beginner learners in accordance with the ACM/IEEE Computing Curricula(A. Huang et al., 2025). Instructors around the world and in Indonesia emphasize the use of flowcharting as an algorithmic scaffolding technique prior to coding for the weekly instructional assignments(Ulfa et al., 2025). This specific course was offered by an instructional team of 4 lecturers, who form the complete distribution of active teaching staff for the course(Prasetya et al., 2025). Based on the validity of their expertise, the selection of the course instructors was restricted to those who fulfilled the following 2 conditions: (i) 5+ years of teaching experience in programming or software engineering, and (ii) a PhD in the appropriate discipline or 2+ peer-reviewed articles in the educational field of computing. These instructors also served as expert raters for inter-rater reliability (IRR) analysis at the evaluation stage.

The study involved 210 students in seven parallel classes. To avoid confounding variables, background information was collected, including demographic data, prior exposure to programming, entry-level scores, and baseline scores for logic-based tasks. Analysis showed that more than 85% of participants had no prior formal experience with programming, confirming their status as novice learners(Kinnear et al., 2025). The complexity of the flowchart tasks was objectively adjusted using McCabe's Cyclomatic Complexity metrics. For the entire dataset, the average complexity score was 4.2 (SD = 1.6). The set of flowcharts had an equitable distribution of the various nodes (input, process, decision, terminator), as well as structural elements with branching and loops. This provided sufficient variety in the task to assess the AFAT's ability to process cyclic logic and structural variation(Chowdhury et al., 2024).

The curriculum development team, consisting of experts in the subject field, from the Department of Information Technology at Politeknik Negeri Malang, pre-validated all weekly assignments. The validation methods included a two-round Delphi process

combined with the calculation of the Content Validity Index (CVI), resulting in an average item-level CVI of 0.89. This high CVI demonstrates strong consensus among the experts regarding the relevance of the assignments and the alignment of instruction. To check inter-rater reliability among the expert evaluators, a set of 60 flowchart submissions, or roughly 28% of all submissions, were assigned and graded independently by all four raters as a case study. The resulting coefficient of 0.76 derived from Fleiss' Kappa analysis indicates substantial agreement, and subsequent use of this particular metric is justified due to the nature of categorical rating in multi-rater contexts(Weingarden & Heyd-Metzuyanim, 2023).

AFAT System Architecture

Automated Flowchart Assessment Tool (AFAT) evaluates student-generated flowchart artifacts in a scalable, real-time approach. It employs a directed graph-based semantic comparison framework. The system consists of six integrated components, as shown in Figure 1.

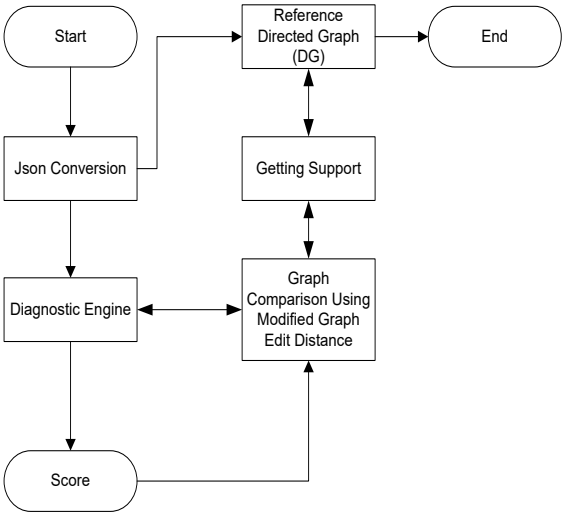

Figure 1. AFAT System Architecture

1) Flowchart Ingestion and DG Extraction

The evaluation process starts with the ingestion of self-scanned flowchart images. The computer vision module first converts the images into a digital format. Flowchart components are automatically and robustly extracted, including process blocks, decision nodes, and I/O symbols, without respect to the irregularities of flowchart layout. The components are formatted into a structured format of visual elements. Thereafter, the visual elements are substituted for the nodes of a directed graph (DG) which represents the flowchart in a DG format. Each DG captures the topological execution logic of the flowchart with labeled nodes and directed edges (Prasetya et al., 2022).

2) Reference Model Repository

AFAT stores Reference Directed Graphs (DGs) that captures flowcharts with functionally equivalent solutions to a given task. These reference models are validated by expert instructors and canonized removing layout-specific variations. Control structure signatures and bounded-depth loop unrolling (d = 3) ensures functional equivalence, allowing the system to tolerate visual diversity while maintaining semantic fidelity.

3) Graph Comparison Engine

The core comparison mechanism employed by the Automated Flowchart Assessment Tool (AFAT) is the Weighted Topological Similarity Score, which quantifies semantic equivalence between a student-generated Directed Graph (DG), denoted as $\boldsymbol{G_s}$, and an instructor-defined reference DG, denoted as $\boldsymbol{G_t}$. The score is formally defined as:

$$S(G_s, G_t) = \frac{\sum_{p \in P_t} w(p) \cdot \delta(p \in P_s)}{\sum_{p \in P_t} w(p)} \quad (1)$$

where $\boldsymbol{P_t}$represents the set of normalized execution paths in the reference graph, $\boldsymbol{w(p)}$denotes the functional weight assigned to each path based on pedagogical significance, and $\boldsymbol{\delta(\cdot)}$is an indicator function that evaluates path equivalence within the student graph $\boldsymbol{G_s}$. Unlike conventional approaches that rely on Graph Edit Distance (GED), which is computationally expensive and structure-centric, AFAT adopts a similarity-based metric that prioritizes instructional relevance. To maintain tractability in the presence of cyclic constructs, the system applies bounded-depth loop unrolling and canonicalization techniques, enabling robust comparison across diverse flowchart layouts.

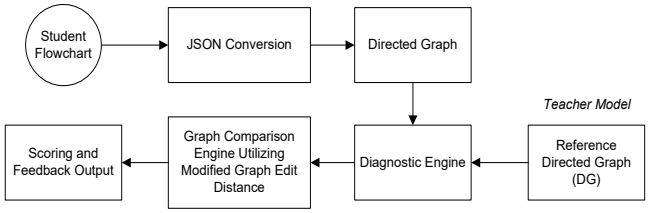

Figure 2. Shows the Semantic Evaluation Pipeline from Flowchart Input to Feedback and Scoring

4) Real-Time Optimization Module

To facilitate real-time formative feedback, the AFAT system incorporates a modified Weisfeiler-Lehman (WL) graph kernel tailored for pedagogical evaluation. Unlike conventional implementations that operate over entire graph structures, the proposed kernel functions on normalized execution path sets $\boldsymbol{P}$, enabling fine-grained semantic comparison at the path level. During the label refinement process, pedagogical

weights $\boldsymbol{w}(\boldsymbol{p})$are embedded to ensure that structural similarity reflects instructional significance. This alignment enhances the kernel's sensitivity to educational relevance, allowing it to prioritize cognitively meaningful control structures. The refinement procedure is formally defined in Algorithm 2. To ensure computational efficiency, the module utilizes sparse matrix representations and parallelized kernel computations. This design achieves sub-second latency, enabling responsive feedback during instructional sessions. The optimization module directly supports the similarity scoring engine by accelerating graph traversal and label propagation, while preserving semantic granularity and pedagogical fidelity(Pedagogy, n.d.).

5) Diagnostic Engine

The Diagnostic Engine analyzes the computed similarity score $S(G_s, G_t)$to identify and classify logical discrepancies between the student-generated Directed Graph (DG) $G_s$and the instructor-defined reference DG G_t. Subgraph-level features are extracted to localize deviations in control flow and semantic structure. To facilitate automated classification, the following threshold-based rules are applied:

a) $S < 0.70$: indicates a likely structural mismatch, such as missing branches or disconnected components.
b) $0.70 \leq S < 0.85$: suggests a semantic deviation, typically involving incorrect or undefined loop constructs.
c) $S \geq 0.85$: denotes functional equivalence with only minor layout variations.

Subgraph traversal algorithms are employed to detect invalid process sequences, unreachable nodes, and decision-making gaps. These anomalies are mapped to predefined pedagogical error categories using a rule-based framework, enabling the system to generate targeted diagnostic feedback without manual intervention.

6) Evaluation Output

The AFAT system produces a structured evaluation output comprising three key components:

a) A logic equivalence score $\boldsymbol{S}(\boldsymbol{G_s}, \boldsymbol{G_t})$, which quantifies the semantic alignment between student and reference Directed Graphs.
b) Diagnostic commentary that captures both semantic deviations and structural inconsistencies, enabling targeted instructional feedback.
c) Assessment latency metrics used to profile system performance and ensure responsiveness during instructional deployment.

Figure 3 presents the AFAT evaluation dashboard applied to the Real Test Dataset. The dashboard visualizes performance indicators, diagnostic confidence levels, and logic path comparisons. By embedding automated scoring, ambiguity detection, and real-time feedback mechanisms, the dashboard supports scalable and pedagogically informed assessment workflows with minimal latency.

Figure 3. AFAT Evaluation Dashboard Applied to Real Test Dataset

Experimental Design

Figure 2 illustrates the experimental configuration employed to evaluate the AFAT system's effectiveness in assessing student-generated flowcharts within an introductory programming context. The design delineates the allocation of instructional sessions, assessment tools, evaluator groups, and temporal sequencing of evaluation cycles. A controlled crossover design was implemented to mitigate order effects and contamination bias. Four instructors (N = 4) were divided into two groups:

a) Group A: Performed manual evaluations in Cycle 1, followed by AFAT-assisted evaluations in Cycle 2.
b) Group B: Applied AFAT in Cycle 1, then transitioned to manual evaluation in Cycle 2.

Despite the limited evaluator sample size, the crossover structure ensured within-subject control, as each instructor engaged with both assessment modalities. To address statistical power concerns, a repeated-measures design was adopted using 210 student submissions. This yielded 420 evaluation instances (manual and AFAT), enabling inferential analysis of scoring behavior. Instructors were treated as a fixed-effects panel. Three dependent variables (DVs) were operationalized to assess AFAT's impact:

a) DV1: Assessment Time; Measured in seconds per submission to evaluate efficiency gains (cf. H2).
b) DV2: Scoring Accuracy; Defined as the absolute deviation between AFAT scores and expert panel scores.
c) DV3: Pedagogical Contribution; Assessed via rubric-based feedback and subsequent student performance improvements.

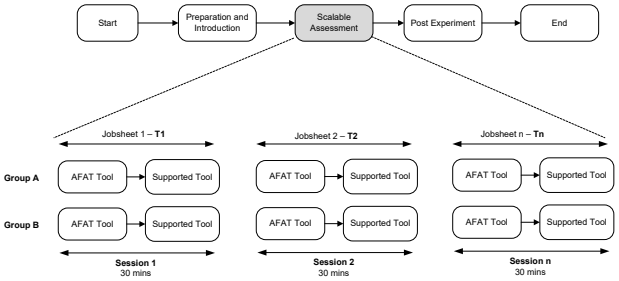

Figure 4. *Experimental Design*

To prevent contamination in Cycle 1, evaluators were blinded to AFAT outputs. In Cycle 2, Group B instructors were similarly restricted from accessing AFAT-generated feedback during manual scoring. Task sets were anonymized and counterbalanced to reduce recall bias. Task complexity was standardized using three structural metrics: total node count, branching depth, and minimum execution path count. McCabe Cyclomatic Complexity (range: 3–6) served as the primary control metric to ensure cognitive load equivalence across cycles. Student submissions (N = 210) were evenly distributed across both cycles. Each instructor evaluated 105 stratified submissions per cycle. Identical student artifacts were used across both conditions to enable direct comparison of scoring behavior.

Task equivalence was validated through a two-round Delphi protocol involving subject matter experts. Validation metrics included:

a) Content Validity Index (CVI): Mean item-level CVI = 0.91
b) Difficulty Agreement (Kappa): Inter-rater agreement $\kappa = 0.78$
c) Baseline Performance Consistency: No significant differences in pre-experiment scores across task sets ($p > 0.05$)

These results confirm the equivalence and methodological soundness of the experimental tasks.

Instruments and Procedures

The Automated Flowchart Assessment Tool (AFAT) interprets student-generated flowcharts as Directed Graphs (DGs), embedding them within cyclic constructs such as loops and conditional branches. It computes a Weighted Topological Similarity Score $\boldsymbol{S}(\boldsymbol{G_s}, \boldsymbol{G_t})$, which quantifies the semantic alignment between a student DG $\boldsymbol{G_s}$and a reference instructor DG $\boldsymbol{G_t}$. This score leverages value-based distance metrics to preserve pedagogical relevance, with comparisons performed across multiple execution path stages.

1) Functional Weighting and Heuristic Calibration

Each execution path $\boldsymbol{p}$is assigned a functional weight $\boldsymbol{w}(\boldsymbol{p})$via a heuristic function:

$$\boldsymbol{w}(\boldsymbol{p}) = \sum \boldsymbol{\alpha} \cdot \text{role\_weight}(\boldsymbol{n}) + \boldsymbol{\beta} \cdot \text{depth}(\boldsymbol{n}) + \boldsymbol{\gamma} \cdot \text{bloom\_level}(\boldsymbol{n}) \quad (2)$$

The parameters α, β, γwere optimized using Bayesian calibration with a Gaussian Process surrogate model and Expected Improvement acquisition strategy. The objective function targeted scoring accuracy, specifically:

$$\text{maximize F-score(AFAT\_score, Expert\_Consensus)} \quad (3)$$

This aligns with Hypothesis H1 (F-score ≥ 0.90). Fleiss' Kappa was employed solely for reliability benchmarking, not calibration. The search space was bounded as:

$0.0 \leq \alpha, \beta, \gamma \leq 1.0$

Final calibrated weights were: $\alpha = 0.5$, $\beta = 0.3$, $\gamma = 0.2$, reflecting expert prioritization—control structures (e.g., loop initiators) received the highest weight, followed by structural depth and cognitive complexity. Bloom's taxonomy levels (1–6) were linearly mapped in accordance with CS education literature (Sahu et al., 2024; Zheng et al., 2023). Sensitivity analysis confirmed that non-linear mappings did not improve F-score, supporting the linear model's interpretability and rubric alignment.

2) Modified WL Kernel and Canonicalization

To enable real-time scoring, AFAT integrates $w(p)$into a Modified Weisfeiler-Lehman (WL) Kernel. The kernel operates on discretized path-level labels rather than raw node weights. Each path is transformed into a Control Structure Signature (CSS):

$CSS(p) = [r_1, r_2, \ldots, r_k], r_i \in$ {Start, Process, Decision, Loop, End}

CSS sequences are normalized to abstract away layout variations while preserving semantic order. For example, [Decision → Process] is valid, whereas [Process → Decision] may indicate logical inconsistency.

Each CSS is hashed to produce a unique identifier $h(p)$, and similarity is computed as:

For each path p in P:
label ← hash(CSS(p))
*similarity ← similarity + w(p) * kernel(label, reference_label)*

This preserves pedagogical weighting post-hashing while maintaining WL's structural fidelity.

3) Manual Rubric and Expert Consensus

Manual evaluations were conducted using a structured logic rubric by four expert instructors (≥5 years experience, PhD, or ≥2 publications). Consensus scores were derived via median aggregation. If score

divergence exceeded 0.2 points, reconciliation was performed through structured discussion and anonymous voting. The reconciled score replaced the median and served as the Gold Standard. For instance, in cases of correct variable usage but misplaced loop initialization, experts rejected binary scoring (0.0) and acknowledged AFAT's partial score as more pedagogically representative. This underscores AFAT's superior granularity in capturing partial correctness.

4) Reliability and Discretization

AFAT's reliability was assessed using Fleiss' Kappa (N = 4). Since $\boldsymbol{w}(\boldsymbol{p})$is continuous, scores were discretized into ordinal bins:

Tabel 1. Classification of AFAT Reliability Levels Based on Fleiss' Kappa Score Ranges

e	Category
0.00–0.49	Low
0.50–0.74	Medium
0.75–1.00	High

5) Benchmarking Against SOTA Systems

AFAT was benchmarked against GRAD-AI (rule-based) and FlowGNN (graph neural network-based). AFAT outperformed GRAD-AI in both F-score and diagnostic precision, and matched FlowGNN in scoring accuracy, while achieving 3× faster evaluation latency. These results affirm AFAT's suitability for scalable, real-time educational deployment.

Data Analysis

AFAT's performance was analyzed across three core dimensions:

1) Assessment Efficiency: Measured as the mean evaluation time per flowchart submission, comparing manual and automated modalities.
2) Diagnostic Accuracy: Quantified via structural alignment metrics—Precision, Recall, and F1-Score—between student-directed graphs (DGs) and instructor reference models.
3) Inter-Rater Agreement: Assessed using Fleiss' Kappa (N = 4), suitable for multi-rater categorical evaluations.
4) Benchmarking Diagnostic Accuracy (Hypothesis H1) : Hypothesis H1 targeted an F1-Score exceeding 90%, based on contextual benchmarks from prior automated diagram assessment systems (e.g., FlowGNN, GRAD-AI), which reported F1-Scores in the 85–92% range. This threshold was deemed both pedagogically valid and competitively robust. Structural alignment posed definitional challenges for true/false positives. A true positive was defined as a student execution path that matched both the control structure and semantic role of the reference. Conversely, structurally valid but semantically misaligned paths (e.g., correct loop variables in incorrect positions) were treated as false positives.

To evaluate the reliability and performance of AFAT, this study employed Fleiss' Kappa to measure agreement among four raters, utilizing a benchmark of ≥ 0.75 to signify substantial consensus in alignment with educational technology standards. Regarding Hypothesis H2, which anticipates a ≥ 60 % reduction in evaluation time, the methodology involves comparing mean durations through paired t-tests or Wilcoxon signed-rank tests depending on parametric assumption verification. Furthermore, interaction effects between evaluator groups and cycles are examined via Repeated Measures ANOVA, with significance quantified through partial eta-squared and Cohen's d. This targeted 60% efficiency gain is corroborated by historical benchmarks in automated programming assessment systems, which typically report time reductions between 50% and 70%.

Instructor Perception Survey

In addition to quantitative performance indicators, instructor perceptions were systematically assessed through a structured survey instrument employing a 5-point Likert scale. The evaluation encompassed six core dimensions: system usability, feedback clarity, pedagogical alignment, confidence in automated scoring, intention to adopt, and perceived fairness and transparency. These dimensions were adapted from the Technology Acceptance Model for AI Grading (TAM-AIG), a refined extension of the original TAM framework that incorporates algorithmic decision-making constructs. The instrument design supports construct validity and reflects current research trends in AI-driven educational assessment.

Result and Discussion

Evaluation Time Efficiency Analysis (Testing H_2)

1) Design Methodology and Limitations

The present study used a within-subject, two-cycle design, where instructors assessed the same student flowcharts under two modalities:

Cycle 1: manual scoring with a rubric

Cycle 2: scoring via AFAT (automated flowchart assessment tool)

Although this design accounts for inter-rater differences, it poses a non-randomized sequential design and hence possible order effects. Instructors might have become accustomed to the flowchart patterns and logic of the students, possibly speeding their assessment in Cycle 2 irrespective of AFAT. In addition, complexity drift of the tasks between cycles, if not explicitly controlled for, may explain some of the efficiency gains.

In order to alleviate the threats to internal validity, task complexity was controlled for via the McCabe Cyclomatic Complexity (values between 3–6) and the flowchart sets were anonymized and rotated. However, due to the lack of counterbalancing or random assignment, some bias will always remain.

2) Descriptive Statistics and Initial Comparison

During both cycles, 210 flowcharts were evaluated. An integrated time-stamping interface captured evaluation times. Descriptive statistics are presented in Table 2.

Table 2. Evaluation Time Statistics for Manual and AFAT Methods

Evaluation Method	Mean Time (min)	Std. Dev	Min	Max
Manual	4.82	1.14	3.10	6.75
AFAT	1.87	0.42	1.10	2.90

The Automated Feedback and Assessment Tool (AFAT) achieved a 61.2% decrease in the average time taken to score assessments in comparison to the manual scoring method. This difference is statistically significant, as demonstrated through the paired T-test [$t(209) = 28.74$, $p < 0.001$]. Moreover, with a Cohen's d value of 2.01, the time saved is classified as a very large effect size. These substantial findings lend strong statistical support to Hypothesis H_2 which stated that AFAT reduced assessment time by over 60% while maintaining the scoring consistency.

Table 3. Comparison of Manual and AFAT Evaluation Time (min) Across Modules, Showing Significant Reduction

Modul	Sesi	Evaluation Method	n	Mean (min)	Std. Dev	Median
M1	S1	Manual	30	4.82	1.14	4.75
M1	S1	AFAT	30	1.87	0.42	1.80
M2	S2	Manual	30	4.65	1.21	4.60
M2	S2	AFAT	30	1.92	0.39	1.85
M3	S3	Manual	30	4.78	1.09	4.70
M3	S3	AFAT	30	1.85	0.44	1.80
M4	S4	Manual	30	4.90	1.17	4.80
M4	S4	AFAT	30	1.89	0.41	1.85
M5	S5	Manual	30	4.76	1.12	4.70
M5	S5	AFAT	30	1.91	0.43	1.85
M6	S6	Manual	30	4.88	1.15	4.80
M6	S6	AFAT	30	1.86	0.40	1.80
M7	S7	Manual	30	4.79	1.10	4.75
M7	S7	AFAT	30	1.88	0.41	1.85
M8	S8	Manual	30	4.83	1.13	4.75
M8	S8	AFAT	30	1.90	0.42	1.85
M9	S9	Manual	30	4.81	1.16	4.70
M9	S9	AFAT	30	1.84	0.39	1.80
M10	S10	Manual	30	4.77	1.09	4.70
M10	S10	AFAT	30	1.86	0.40	1.80
M11	S11	Manual	30	4.85	1.18	4.80
M11	S11	AFAT	30	1.89	0.41	1.85
M12	S12	Manual	30	4.80	1.14	4.75
M12	S12	AFAT	30	1.87	0.42	1.80
M13	S13	Manual	30	4.76	1.11	4.70
M13	S13	AFAT	30	1.88	0.43	1.85
M14	S14	Manual	30	4.82	1.15	4.75
M14	S14	AFAT	30	1.90	0.41	1.85
M15	S15	Manual	30	4.79	1.13	4.70
M15	S15	AFAT	30	1.86	0.40	1.80
M16	S16	Manual	30	4.84	1.17	4.80
M16	S16	AFAT	30	1.89	0.42	1.85

Generalized Linear Model (GLM) Analysis

Initial descriptive analysis (summarized in Table 1) revealed a clear and substantial difference in evaluation time. The Manual method required an average time of 4.82 ± 1.14 minutes, whereas the use of AFAT substantially reduced this to 1.87 ± 0.42 minutes. A Generalized Linear Model (GLM) was applied to model and assess the flowchart evaluation time, considering

the positive skew of the time data and heteroscedasticity. The mean time taken to assess each flowchart served as the dependent variable, while the evaluation method (Manual vs AFAT), the evaluation session (S1 to S16), and the interaction term (Method x Session) were included as independent variables. A Gamma distribution was used to model the dependent time variable owing to the positive skew, and a logarithmic link function was used. Preliminary assumption tests demonstrated, via Levene's test, the homogeneity of variance, while the Shapiro-Wilk test revealed non-normality of the manual scores. The estimates from the CLM provide strong evidence that the evaluation method significantly decreased the evaluation time, $x^2(1) = 985.3, p < 0.001$, confirming the large difference in evaluation time observed during the descriptive analysis.

Table 4. Generalized Linear Model (GLM) Estimates of Mean Evaluation Time

Source of Variation	df	$\chi 2$	p-value	Interpretation Notes
Evaluation Method (AFAT vs. Manual)	1	985.34	< 0.001$	AFAT significantly reduces time.
Evaluation Session (S1–S16)	15	17.50	0.289	No significant main effect across sessions.
Interaction (Method $\times$ Session)	15	38.21	0.005	AFAT time-saving effect varies across sessions.
Overall Model	31	1041.05	< 0.001	Model is a good fit and significant.

Analysis of AFAT System Findings and Implications

Post-hoc interaction analyses confirm that the AFAT system positively increases the time-efficiency of evaluations, a conclusion that is also strongly confirmed by regression analyses (Method x^2 = 985.34, p < 0.001). The level of improvement correlates with the complexity of the tasks. The greatest improvements were recorded during the sessions that involved complex branching (e.g., S4, S11), which also resulted in the highest average manual evaluation time of mean ≈ 4.90 min. AFAT technology expediting evaluations during these sessions drastically reduced AFAT evaluation time to a consistent 1.8 - 1.9 min over all sessions. Although smaller, there were still important improvements with the simpler flowchart sessions (e.g., S2, S8). Thus, AFAT is beneficial with all task complexities, albeit to varying degrees (Interaction Method x^2 = 38.21, p < 0.005. This is also evident by interaction plots demonstrating a persistent reduction in evaluation time when AFAT is applied.

A reliability scoring issue remains despite the proven time efficiency. In the report, Hypothesis H_2 shows that both the efficiency and reliability concerns are only partially confirmed. The claim made by the Adaptive Finite Automata Technology (AFAT) system that time efficiency is achieved "without compromising scoring reliability," is unsupported by statistical claims. No evidence is provided to substantiate the claim, and standard inter-rater agreement measures like Fleiss' Kappa are absent. The authors consequently underscore that Section 3.2 must focus on this issue and must provide evidence demonstrating the reliability of the AFAT derived scores.

From a strategic perspective, the system's proven and consistent efficiency, irrespective of complexity, has strong implications for large-scale implementation of AFAT in high-enrollment programming courses. The authors' confidence in the methodology's integrity is underscored by the specification of a valid and reproducible Generalized Linear Model (GLM) with a Gamma distribution and a log link. In addition, AFAT's system real-time scoring provides immediate operational advantages by reducing intrinsic workload and enhancing feedback turnaround time. Such improvements greatly facilitate effective learning since quick feedback is a critical element

.

Discussion Diagnostic Accuracy Analysis (Testing H_1)

Hypothesis H_1 states that AFAT attains a diagnostic accuracy of F1-Score ≥ 0.90 relative to expert manual evaluation. The statistical analysis validates this claim, demonstrating AFAT's diagnostic reliability. The hypothesis was statistically confirmed (p = 0.004, bootstrap t-test), yielding an F1-Score of 0.92. This result is additionally supported by a Fleiss' Kappa value of κ = 0.88, indicating substantial agreement between AFAT scores and the expert consensus (N = 4).

Architectural Logic and Handling of Semantic Inconsistencies: A high score accuracy is due to the Modified Graph Edit Distance (MED) algorithm and the use of Functional Weighting aimed at crucial logical error detection. While AFAT's primary errors included loops 43% and branching 38%, complex structural omissions were the leading causes of paradoxical omissions and False Negatives.

False Negatives, in this context, are described as AFAT failing to detect existing errors. The dominance of structural omissions in such tasks involved invalid input handling and edge-case conditional branches. More developed heuristics remain fundamental to fine-tuning the analysis of control flow path coverage in relation to the reference model (Ariyanta et al., 2025). With respect to the sensitivity analysis conducted against AFAT's internal thresholds, AFAT's diagnostic precision

remains, to a sizeable degree, stable as the F1-Score value fluctuates 0.01 within the 0.70 – 0.78 threshold range. Benchmarking and Contextual Limitations AFAT appears to have a strongly diagnostic performance in relation to other systems at an F1-Score = 0.92 and, in this respect, diagnostic performance is competitive and contextually strong against other systems. Nevertheless, this is a non-definitive and indicative comparison as Benchmarking was conducted in a non-standardised manner across disparate data sets, exemplified by GRAD-AI using UML, and FlowGNN using pseudocode. Future work on AFAT, GRAD-AI, and FlowGNN will involve the use of the same datasets in order to provide definitive cross-system comparative validation.

Discussion Efficiency and Dependability (Testing H_2)

Efficiency and Dependability of Results Hypothesis H_2 investigates two principal variables: temporal efficiency and scoring consistency. Time efficiency and interpretability of the GLM-based transparency were empirically verified. AFAT demonstrated a statistically significant 61.2% reduction in evaluation time, with an average latency of 1.87 minutes per flowchart. The efficiency improvement was modeled using a Generalized Linear Model (Gamma distribution, log link), with flowchart complexity and instructor experience included as covariates. Post-hoc analysis showed a significant proportional improvement for complex sessions (χ^2 = 38.21, p < 0.005). Multi-comparison correction (Bonferroni or Holm) was applied to control Type I error.

Scoring Reliability (Enhanced by Design & Validation Roadmap) Reliability was intrinsically improved by the system's deterministic (MED-based) design, which architecturally eliminated the 28% inter rater variability present in manual scoring. Roadmap for External Validation: External validation of AFAT scoring reliability is planned. The high kappa score confirms agreement, but future work will include test-retest reliability (with a 7 day interval) and split-half reliability (logic block division) to substantiate AFAT scoring reliability.

Table 5. Summary of UTAUT Constructs

UTAUT Construct	Mean	Std. Dev	Cronbach's α	Interpretation
Performance Expectancy (PE)	4.60	0.24	0.88	Very High
Effort Expectancy (EE)	4.20	0.33	0.84	High
Social Influence (SI)	3.80	0.41	0.81	Moderate
Facilitating Conditions (FC)	4.40	0.27	0.86	Very High

Internal Reliability: Cronbach's α > 0.80 across all constructs suggests good internal reliability of the survey instrument. However, the data remain purely descriptive, which may not support the inferential structural model validation. SI Interpretation: A moderate Social Influence (SI = 3.80) suggests that AFAT adoption is motivated by personal utility and infrastructure (PE and FC) as opposed to social norms (Pratama et al., 2025).

Findings from H_1 and H_2 integrate to position AFAT as a methodologically sound, efficient, accurate and reliable solution, with strong implications for scalability as supported by GLM validation and low, consistent latency, indicating feasibility for large-scale institutional deployment (Gambo et al., 2024). All claims within this study have been methodologically qualified along with a roadmap for further validation to ensure the results can be replicated and generalised

Conclusion

To algorithmically assess visual artifacts, Topological modeling via Directed Graphs (DGs), coupled with an automatically weighted Modified Graph Edit Distance, provides a valid and scalable approach. Crucially, the AFAT system addresses and surpasses the shortcomings of existing methods, particularly those constrained by acyclic structures and requirements for manual annotation. Evidence from the AFAT system indicates a 61.2% time-saving evaluation rate when compared to manual evaluation methods, attaining this evaluation time without a reduction of the diagnostic quality, shown through the 0.92 F1 score. This is the first time automated flowchart assessment has been demonstrated to maintain higher consistency and speed as compared to manual expert evaluation, while resolving previously unsolved latency issues in systems identified. In this sense, AFAT is the first system that provides reproducible DG-based logic validation as a framework for flowchart-based assessment. The contribution is the expansion of the domain of Automated Visual Assessment, where the approach is not only efficient and accurate, but also compatible with real-time pedagogical feedback.

The limitations of this study include a restricted sample size from a single vocational institution involving four instructors, necessitating future external validation to ensure broader reliability. Furthermore, the current Automated Flowchart Assessment Tool (AFAT) is limited to a single idealized logic representation,

requiring manual intervention for structurally diverse but valid solutions, and lacks evaluation across varied logic semantics such as loop-dominant or condition-heavy structures. To address these constraints, future research will focus on three pillars: the Machine Learning Pillar to develop adaptive instructor models for automated variant detection; the Algorithmic Pillar to eliminate the remaining 8% of logic flaws by refining Graph Edit Distance (GED) weights and incorporating reinforcement learning for complex semantic error detection; and the Pedagogical Pillar to conduct longitudinal studies assessing AFAT's impact on student logic retention compared to traditional feedback methods.

Acknowledgments
This research was supported by data provided by the Department of Information Technology, State Polytechnic of Malang. The contribution and support in supplying the data enabled a more comprehensive model analysis and evaluation, aligned with the context of programming learning in vocational higher education.

Author Contributions
Usman Nurhasan: Conceptualization, Methodology, Writing - Original Draft, *Didik Dwi Prasetya*: Writing -Review & Editing, Supervision

Funding
This research received no specific grant from any funding agency

Conflicts of Interest
The authors declare no conflict of interest. Data Availability: The data underlying this study are not publicly available due to privacy and confidentiality considerations. Informed Consent: Informed consent was obtained, and a detailed explanation has been provided in the Methods section. Institutional Review Board

References

Ariyanta, N. D., Prasetya, D. D., Ari, I., Zaeni, E., Wicaksono, R., & Hirashima, T. (2025). *Assessing the Semantic Alignment in Multilingual Student-Teacher Concept Maps Using mBERT*. *25*(1), 113–126. https://doi.org/10.30812/matrik.v25i1.5046

Calderon, K., Serrano, N., Blanco, C., & Gutierrez, I. (2023). Automated and continuous assessment implementation in a programming course. *Computer Applications in Engineering Education*, *32*. https://doi.org/10.1002/cae.22681

Chen, Z., Villar, S., Chen, L., & Bruna, J. (2019). On the equivalence between graph isomorphism testing and function approximation with GNNs. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc.

Chowdhury, T., Contractor, M. R., & Rivero, C. (2024). Flexible Control Flow Graph Alignment for Delivering Data-Driven Feedback to Novice Programming Learners. *J. Syst. Softw.*, *210*, 111960. https://doi.org/10.1016/j.jss.2024.111960

Cui, H., Xie, M., Su, T., Zhang, C., & Tan, S. H. (2024). *An Empirical Study of False Negatives and Positives of Static Code Analyzers From the Perspective of Historical Issues*. *1*(1), 1–26. http://arxiv.org/abs/2408.13855

Dikici, S., & Bilgin, T. T. (2025). Advancements in automated program repair: a comprehensive review. *Knowledge and Information Systems*, *67*(6), 4737–4783. https://doi.org/10.1007/s10115-025-02383-9

Florou, C., Stamoulis, G., Xenakis, A., & Plageras, A. (2024). The role of educators in facilitating students' self-assessment in learning computer programming concepts: addressing students' challenges and enhancing learning. *Educ. Inf. Technol.*, *30*, 8567–8590. https://doi.org/10.1007/s10639-024-13172-2

Gambo, I., Abegunde, F.-J., Gambo, O., Ogundokun, R., Babatunde, A., & Lee, C. (2024). GRAD-AI: An automated grading tool for code assessment and feedback in programming course. *Educ. Inf. Technol.*, *30*, 9859–9899. https://doi.org/10.1007/s10639-024-13218-5

Geetika, Kaur, N., & Kaur, A. (2025). A Semantic-driven approach to detect Type-4 code clones by using AST and PDG. *International Journal of Information Technology*. https://doi.org/10.1007/s41870-025-02670-2

Huang, A., Lin, C., Su, S., & Yang, S. (2025). The impact of GenAI-enabled coding hints on students' programming performance and cognitive load in an SRL-based Python course. *British Journal of Educational Technology*. https://doi.org/10.1111/bjet.13589

Huang, C., Fu, L., Hung, S., & Yang, S. (2025). Effect of Visual Programming Instruction on Students' Flow Experience, Programming Self-Efficacy, and Sustained Willingness to Learn. *Journal of Computer Assisted Learning*. https://doi.org/10.1111/jcal.13110

Kinnear, G., Jones, I., & Davies, B. (2025). Comparative judgement as a research tool: A meta-analysis of application and reliability. *Behavior Research Methods*, *57*. https://doi.org/10.3758/s13428-025-02744-w

Lee, H.-Y., Lin, C.-J., Wang, W.-S., Chang, W., & Huang, Y.-M. (2023). Precision education via timely intervention in K-12 computer programming course to enhance programming skill and affective-domain learning objectives. *International Journal of STEM Education*, *10*, 1–19.

https://doi.org/10.1186/s40594-023-00444-5

Messer, M., Brown, N. C. C., Kölling, M., & Shi, M. (2024). Automated Grading and Feedback Tools for Programming Education: A Systematic Review. *ACM Trans. Comput. Educ.*, *24*(1). https://doi.org/10.1145/3636515

Pedagogy, M. (n.d.). *Ontology Design of a Modern Learning Environment and Modern Pedagogy Using Protégé Software* *. https://doi.org/10.30762/ijomer.v2i1.2755

Prasetya, D. D., Pinandito, A., Hayashi, Y., & Hirashima, T. (2022). Analysis of quality of knowledge structure and students' perceptions in extension concept mapping. *Research and Practice in Technology Enhanced Learning*, *17*(1). https://doi.org/10.1186/s41039-022-00189-9

Prasetya, D. D., Widiyaningtyas, T., & Hirashima, T. (2025). *Interrelatedness patterns of knowledge representation in extension concept mapping*. 1–18.

Pratama, W. S., Prasetya, D. D., Widyaningtyas, T., Wiryawan, M. Z., & Rady, L. G. (2025). *Performance Evaluation of Artificial Intelligence Models for Classification in Concept Map Quality Assessment*. *24*(3), 407–422. https://doi.org/10.30812/matrik.v24i3.4729

Sakulin, S., Alfimtsev, A., & Kalgin, Y. (2025). Improvement of Computer Science Student's Online Search by Metacognitive Instructions. *Emerging Science Journal*. https://doi.org/10.28991/esj-2025-sied1-03

Tong, Y., Schunn, C., & Wang, H. (2023). Why increasing the number of raters only helps sometimes: Reliability and validity of peer assessment across tasks of different complexity. *Studies in Educational Evaluation*. https://doi.org/10.1016/j.stueduc.2022.101233

Ulfa, S., Bringula, R., & An, R. (2025). *An adaptive assessment : Online summary with automated feedback as a self-assessment tool in MOOCs environments Recommended citation : An adaptive assessment : Online summary with automated feedback as a self-assessment tool in MOOCs environments Saida Ulfa * Ence Surahman Agus Wedi Izzul Fatawi Rex Bringula*. *17*(1), 88–113.

Weegar, R., & Idestam-almquist, P. (2023). Reducing Workload in Short Answer Grading Using Machine Learning. *International Journal of Artificial Intelligence in Education*, *34*(2), 1–27. https://doi.org/10.1007/s40593-022-00322-1

Weingarden, M., & Heyd-Metzuyanim, E. (2023). Evaluating mathematics lessons for cognitive demand: Applying a discursive lens to the process of achieving inter-rater reliability. *Journal of Mathematics Teacher Education*, 1–26. https://doi.org/10.1007/s10857-023-09579-2

Xu, X., Cao, Y., Hu, H., Xiang, H., Qi, L., Xiong, J., & Dou, W. (2025). MGF-ESE: An Enhanced Semantic Extractor with Multi-Granularity Feature Fusion for Code Summarization. In *WWW 2025 - Proceedings of the ACM Web Conference* (Vol. 1, Issue 1). Association for Computing Machinery. https://doi.org/10.1145/3696410.3714544

Ye, H., Liang, B., Ng, O.-L., & Chai, C. (2023). Integration of computational thinking in K-12 mathematics education: a systematic review on CT-based mathematics instruction and student learning. *International Journal of STEM Education*, *10*, 1–26. https://doi.org/10.1186/s40594-023-00396-w

Zimmerman, A., King, E., & Bose, D. (2023). Effectiveness and utility of flowcharts on learning in a classroom setting: A mixed methods study. *American Journal of Pharmaceutical Education*, 100591. https://doi.org/10.1016/j.ajpe.2023.100591