

Transfer of Large Language Model Generated Feedback Beyond Assisted Programming Tasks A Secondary Crossover Analysis of ProgFeed

Mushtaq Talib Tally*¹

Iraq Ministry of Education Directorate of Education in Babil, Iraq, Open Educational College
mushtaqtalb@gmail.com

Abstract: Large language models can improve students' performance during programming tasks, but whether such gains transfer to later work completed without artificial intelligence remains uncertain. This study presents a secondary crossover analysis of ProgFeed, an open classroom dataset containing 6693 submissions from 215 consenting students in an introductory Python course. The analysis focused on three counterbalanced intervention periods followed by artificial intelligence free prelabs designed as posttests of the preceding lab. These periods yielded 607 student lab assignments from 211 students, while 89 students completed 190 subsequent posttest observations. Baseline first submission performance was comparable across assigned conditions, and prior feedback assignment did not significantly predict posttest completion. Fixed effects analyses controlling for student and period found no detectable advantage of either test case feedback or natural language feedback in first attempt posttest performance compared with no artificial intelligence feedback. First attempt and eventual full correctness were also not significantly different. The findings suggest a performance transfer gap in which feedback can support debugging within an assisted task without necessarily producing detectable gains on a subsequent independent task. Because the posttests were optional and completion was limited, the estimates should be interpreted cautiously.

Keywords—large language models; programming education; automated feedback; transfer; learning analytics

1. INTRODUCTION

Generative artificial intelligence has rapidly become part of the programming-learning environment. Large language models can explain program behavior, identify likely errors, generate tests, create examples, and suggest complete implementations. These capabilities may reduce barriers for novice programmers and make individualized feedback available at a scale that is difficult to achieve through conventional instruction alone.

However, an important measurement problem follows from these same capabilities: better work produced with an AI system is not necessarily evidence that the learner has acquired a stronger independent skill. In programming, this distinction is especially important because an executable solution may be partly or largely attributable to the external system. The educational outcome of interest should therefore not be limited to whether students solve the assisted task, but whether they can subsequently perform related reasoning without the assistance.

This performance-learning distinction is increasingly visible in empirical research. Bassner et al. [1] reported that AI-supported programming conditions improved task performance while producing no corresponding superiority in conceptual learning in their setting. By contrast, Kestin et al. [2] found substantial learning gains from a carefully engineered AI tutor, suggesting that generative AI is neither intrinsically beneficial nor intrinsically harmful; instructional design and outcome measurement matter.

ProgFeed provides an unusually useful setting in which to investigate this distinction. Heickal and Lan [3] deployed three forms of feedback in an authentic introductory Python course: no AI feedback, LLM-generated failing test cases, and concise LLM-generated natural-language hints. Their analysis of 6,693 submissions found that natural-language feedback was associated with greater eventual correctness within the same lab and faster convergence to correct solutions, whereas test-case feedback produced more heterogeneous results.

The released dataset also contains optional pre-lab assignments. Crucially, the original study states that each pre-lab served simultaneously as a post-test for the previous lab and a pre-test for the upcoming lab, while pre-labs themselves did not contain AI-generated feedback. This design feature creates an opportunity that was not the focus of the original paper: to test whether assignment to an AI-feedback condition in one lab is associated with performance on the subsequent AI-free pre-lab.

Automated feedback has a long history in programming education. Systems have provided compiler explanations, unit-test outcomes, repair suggestions, structural hints, and next-step guidance. Keuning, Jeuring, and Heeren [4] showed that automated programming-feedback systems vary substantially in feedback content, generation method, adaptability, and evaluation quality. Hattie and Timperley [5] likewise argued that feedback effects depend on its level, content, timing, and the gap it helps the learner close.

Recent systematic reviews show rapid growth in generative AI for programming education, including code explanation, formative feedback, debugging support, and

conversational tutoring [6], [7]. Yet the evidence remains heterogeneous because studies differ in learner population, model capability, prompt structure, feedback modality, and outcome measurement. Process-level work also suggests that learning outcomes depend on how learners engage with generated assistance [8].

Natural-language hints may reduce unproductive search while leaving substantial reasoning to the learner. Failing test cases can expose a behavioral discrepancy without diagnosing its conceptual source. For novices, such counterexamples may require debugging schemas that are not yet well developed. This interpretation is consistent with cognitive-load theory [9]. The ICAP framework further predicts stronger learning as activity becomes more constructive and interactive rather than passive [10].

The distinction between assisted performance and transfer is therefore central. The OECD Digital Education Outlook 2026 [11] highlights the risk of conflating task completion with learning in generative-AI environments. ProgFeed is well suited to a preliminary transfer analysis because its pre-labs were designed to function as post-tests of the preceding lab and did not contain AI feedback.

Accordingly, the objective of this study was to test whether the feedback condition assigned in a preceding programming lab predicted subsequent independent performance on the next AI-free pre-lab. Specifically, we examined baseline balance, post-test completion, first-attempt post-test performance, first-attempt and eventual full correctness, and the robustness of the findings across alternative statistical specifications.

2. MATERIALS AND METHODS

2.1 Data source and study design

The analysis uses the public ProgFeed dataset released by the ML4ED group at the University of Massachusetts Amherst under a CC BY 4.0 license. The original classroom study was IRB approved and conducted in an introductory Python course during Fall 2025 [3]. The public consolidated file contains 17,385 graded function/test rows from 215 consenting students and reconstructs 6,693 distinct submission events using student ID, lab, and submission timestamp.

Each consolidated row includes `student_id`, `lab`, `submission_timestamp`, `experimental-group` assignment, `score`, `maximum score`, `pass/fail` status, `feedback type`, `delivered feedback text`, `source file`, `function name`, `code snippet`, and a `function-change` indicator. The public repository also documents the study design, problem statements, and autograders.

The present analysis focuses on lab05, lab06, and lab07 because these labs contain all three assigned feedback conditions in a counterbalanced pattern. The corresponding transfer windows are lab05 to pre-lab06, lab06 to pre-lab07, and lab07 to pre-lab08. Pre-labs did not provide AI-generated feedback and were designed to serve as post-tests for the preceding lab and pre-tests for the next instructional unit [3].

Treatment was defined by the assigned `ai_feedback_type` in the preceding lab: `no_feedback`, `tc` (test-case feedback), or `nl` (natural-language feedback). Assignment rather than actual feedback delivery was analyzed because feedback was only delivered following failing submissions and was therefore endogenous to performance.

TABLE I. TRANSFER WINDOWS USED IN THE SECONDARY ANALYSIS

Intervention period	Assigned feedback conditions	Subsequent AI-free post-test
lab05	No AI / Test case / Natural language	pre-lab06
lab06	No AI / Test case / Natural language	pre-lab07
lab07	No AI / Test case / Natural language	pre-lab08

2.2 Data reconstruction and outcomes

The public CSV stores one row per graded test or function per submission. Submission events were reconstructed by grouping rows with identical `student_id`, `lab`, and `submission_timestamp`. For each event, the score proportion was computed as the sum of points earned divided by the sum of maximum points across all graded rows.

For each intervention lab, baseline performance was defined as the score proportion on the student's earliest submission, which occurs before any feedback generated in response to that submission can influence the submitted code. For each subsequent pre-lab, first-attempt performance was defined using the earliest submission timestamp. Eventual performance was the maximum score proportion observed across that pre-lab.

2.3 Statistical analysis

The dataset has a repeated-measures structure because many students appear in multiple intervention periods. Completion of the subsequent pre-lab was analyzed with a logistic generalized estimating equation (GEE) with exchangeable within-student correlation, feedback condition and period as fixed effects, and no-AI feedback as the reference condition.

Among completed post-tests, the primary first-attempt score analysis used a two-way fixed-effects linear model containing student and period fixed effects, with standard errors clustered by student. This specification exploits within-student changes in assigned condition while controlling for stable individual differences and strong differences in difficulty across the three pre-labs. Coefficients are interpreted directly as score-proportion differences.

Binary full-correctness outcomes were analyzed as fixed-effects linear probability models with student-clustered standard errors. GEE models were used as sensitivity analyses. A second sensitivity analysis replaced the weighted score proportion with the unweighted fraction of graded rows passed. Finally, a complete-case crossover analysis was performed among students who completed all three subsequent pre-labs.

The analysis is retrospective and was not preregistered. P values are two-sided with $\alpha = .05$. Confidence intervals are emphasized because the optional nature of the pre-labs reduces precision and creates potential selection bias. In accordance with the journal template, * denotes $P < .05$ and ** denotes $P < .01$ where applicable.

2.4 Ethics and data availability

This work is a secondary analysis of publicly released de-identified data. The original classroom study was reported as

IRB approved by the dataset authors [3]. The ProgFeed dataset is publicly available under a CC BY 4.0 license.

3. RESULTS

3.1 Intervention structure and baseline balance

The three intervention labs contributed 607 student-lab assignments from 211 unique students: 203 in lab05, 200 in lab06, and 204 in lab07. Across all periods, 203 assignments were no-AI, 202 were test-case, and 202 were natural-language.

One hundred eighty-eight students participated in all three intervention labs. Of these, 187 followed one of three counterbalanced sequences and therefore received each feedback condition exactly once across lab05-lab07: no AI $\rightarrow$ test case $\rightarrow$ natural language ($n = 61$), test case $\rightarrow$ natural language $\rightarrow$ no AI ($n = 64$), and natural language $\rightarrow$ test case ($n = 62$). One student was recorded as no AI in all three periods.

Before feedback could influence the submitted code, first-submission performance was similar across assigned conditions. In a period-adjusted GEE, test-case assignment differed from no AI by -0.012 score-proportion units (95% CI [-0.071, 0.048], $p = .702$), while natural-language assignment differed by -0.022 (95% CI [-0.079, 0.036], $p = .462$).

TABLE II. BASELINE AND SUBSEQUENT PRELAB PARTICIPATION BY ASSIGNED FEEDBACK CONDITION

Prior condition	Assignments	First-submission score M (SD)	Subsequent pre-lab completed	Completion rate
No AI	203	0.811 (0.322)	60	29.6%
Test case	202	0.802 (0.332)	67	33.2%
Natural language	202	0.790 (0.338)	63	31.2%

3.2 Posttest completion and first attempt performance

Across the 607 intervention-period observations, 190 subsequent pre-labs were completed, representing 89 unique students. In the logistic GEE, neither test-case assignment (OR = 1.19, 95% CI [0.95, 1.47], $p = .126$) nor natural-language assignment (OR = 1.04, 95% CI [0.79, 1.35], $p = .796$) significantly predicted completion relative to no AI.

Among the 190 completed post-test observations, unadjusted first-attempt mean score proportions were 0.688 (SD = 0.350) after no AI, 0.696 (SD = 0.353) after test-case feedback, and 0.603 (SD = 0.428) after natural-language feedback. Difficulty varied substantially by period, with

overall first-attempt means of 0.754 after lab05, 0.464 after lab06, and 0.801 after lab07.

The primary two-way fixed-effects analysis found no detectable positive transfer advantage for either AI condition. Relative to no AI, the test-case estimate was +0.008 (SE = 0.075, 95% CI [-0.138, 0.154], $p = .914$), while the natural-language estimate was -0.086 (SE = 0.103, 95% CI [-0.287, 0.115], $p = .401$). The intervals are wide, particularly for natural-language feedback; therefore the results indicate absence of detectable evidence rather than statistical equivalence.

3.3 Full correctness and repeated attempts

TABLE III. DESCRIPTIVE SUBSEQUENT PRELAB OUTCOMES AMONG COMPLETERS

Prior condition	n	First-attempt score M (SD)	First-attempt fully correct	Eventually fully correct	Mean attempts
No AI	60	0.688 (0.350)	50.0%	78.3%	2.75
Test case	67	0.696 (0.353)	52.2%	83.6%	2.54
Natural language	63	0.603 (0.428)	46.0%	82.5%	2.44

First-attempt full correctness did not differ significantly after test-case assignment (+0.004, 95% CI [-0.213, 0.221], $p = .968$) or natural-language assignment (-0.032, 95% CI [-0.253, 0.190], $p = .780$) in the student- and period-fixed-effects model.

Eventual full correctness within the AI-free pre-lab was numerically higher after both AI conditions, but the differences were imprecise and not statistically significant: +0.070 for test-case feedback (95% CI [-0.113, 0.253], $p = .456$) and +0.065 for natural-language feedback (95% CI [-0.102, 0.233], $p = .446$). Mean attempt counts were 2.75, 2.54, and 2.44 after no-AI, test-case, and natural-language assignment, respectively; sensitivity analyses did not identify significant treatment differences.

3.4 Sensitivity analyses

A GEE analysis of first-attempt score produced the same qualitative conclusion. The test-case effect was -0.011 (95% CI [-0.112, 0.090], $p = .831$). The natural-language estimate was -0.114 (95% CI [-0.237, 0.010], $p = .071$).

When first-attempt performance was reconstructed as the unweighted fraction of graded rows passed rather than points earned divided by maximum points, the student-fixed-effects results remained null: +0.010 for test-case feedback (95% CI [-0.137, 0.157], $p = .894$) and -0.081 for natural-language feedback (95% CI [-0.282, 0.121], $p = .433$).

Thirty-six students completed all three subsequent pre-labs, permitting a complete-case crossover comparison. Their raw mean first-attempt scores were 0.685 after no AI, 0.708 after test-case feedback, and 0.574 after natural-language feedback. The treatment coefficients remained non-significant after student and period adjustment.

TABLE IV. PRIMARY AND SENSITIVITY ESTIMATES FOR SUBSEQUENT ARTIFICIAL INTELLIGENCE FREE FIRST ATTEMPT SCORE

Model	Test case vs no AI	Natural language vs no AI
Student + period fixed effects	0.008 [-0.138, 0.154], $p = .914$	-0.086 [-0.287, 0.115], $p = .401$
GEE, period adjusted	-0.011 [-0.112, 0.090], $p = .831$	-0.114 [-0.237, 0.010], $p = .071$
Unweighted pass fraction, student FE	0.010 [-0.137, 0.157], $p = .894$	-0.081 [-0.282, 0.121], $p = .433$

Note: Values in brackets are 95 percent confidence intervals. * $P < 0.05$; ** $P < 0.01$.

4. DISCUSSION

The central finding of this secondary analysis is a dissociation between within-task debugging success reported in the original ProgFeed study and subsequent AI-free performance in the course's post-test pre-labs. Heickal and Lan [3] reported that natural-language feedback was associated with more than twice the odds of eventual correctness within the assisted lab and a modestly faster rate of convergence. In the present reanalysis, however, assignment to natural-language feedback did not predict higher first-attempt performance on the next pre-lab, and neither AI condition produced a statistically detectable advantage on first-attempt or eventual full correctness after adjustment for period and student.

This should not be interpreted as a contradiction. The two analyses answer different questions. Within-lab success asks whether feedback helps students solve the current problem. The post-test analysis asks whether the experience leaves the student better prepared for a later related task without generative AI. A feedback system can succeed on the first criterion without producing a detectable effect on the second.

The result therefore reinforces a broader methodological point in generative-AI research: the artifact produced during AI use is a joint product of learner and system. When the system directly contributes error diagnosis or solution-relevant information, improvement in that artifact cannot be treated as equivalent to independent learning.

The point estimate for natural-language feedback was negative in the first-attempt post-test models, but the confidence interval was wide and the complete-case sample was small. It would therefore be inappropriate to conclude that natural-language hints harmed learning. The defensible conclusion is narrower: the dataset provides no detectable evidence that the within-lab performance advantage transferred to the subsequent optional AI-free pre-lab.

Test-case feedback showed a similar pattern. The original paper reported heterogeneous within-lab effects that depended strongly on whether the generated test was valid [3]. In the present analysis, prior test-case assignment produced first-attempt post-test scores that were essentially indistinguishable from no AI. Presenting a counterexample therefore does not automatically create transferable debugging knowledge, particularly when learners may not yet possess the conceptual schema needed to interpret the case.

The results also illustrate why period adjustment is essential in educational trace data. Raw performance differed sharply across pre-labs, with average first-attempt score near 0.46 in one period and near 0.80 in another. A naïve comparison of treatment-group means without accounting for task period could therefore be seriously misleading. The counterbalanced within-student structure of lab05-lab07 is a valuable feature of ProgFeed because it permits stronger control of stable learner differences than a simple cross-sectional analysis.

4.1 Implications for educational artificial intelligence design

An educational AI system should not be optimized solely for immediate correction rate. If a hint allows the learner to repair the current program but contributes little to later independent performance, the system may be functioning more as a debugging assistant than as a tutor.

Future systems should incorporate deliberate transfer mechanisms. Examples include asking learners to explain why a hint applies, requiring them to predict behavior before receiving another cue, fading support across similar problems, and scheduling subsequent AI-free retrieval or debugging tasks. These strategies preserve the productivity benefits of AI assistance while creating opportunities for learner-generated reasoning.

Evaluation should likewise contain at least one outcome measured after assistance is removed. Immediate correctness, time-to-solution, satisfaction, and perceived helpfulness remain useful, but they answer different questions from retention and transfer.

4.2 Methodological implications and limitations

The analysis demonstrates the value of secondary use of fine-grained educational datasets. The original study was designed primarily around feedback trajectories within labs, yet its pre-lab structure permits a distinct research question about transfer. Making de-identified traces, assignment labels, and timestamps available enables this form of cumulative science.

Secondary analysis must nonetheless respect the original design. Because feedback delivery occurred only after failure, comparing students who actually saw feedback with those who did not would condition on a post-assignment performance event. The present analysis therefore uses assigned condition as the exposure and treats optional post-test completion separately.

Several limitations constrain interpretation. The subsequent pre-labs were optional extra-credit activities: only 190 of 607 intervention-period observations had a completed post-test, and only 36 students completed all three transfer windows. Although assigned feedback condition did not significantly predict overall completion, conditioning the score analysis on completion can still introduce selection bias.

The pre-labs were course activities rather than psychometrically equated transfer tests. They were explicitly intended to serve as post-tests for the preceding lab, but problem content and difficulty differed substantially across periods. Period fixed effects address mean difficulty differences but cannot make tasks identical.

The analysis estimates the effect of assigned feedback condition rather than feedback actually received. This intention-to-treat approach avoids selection on failure-triggered delivery but dilutes the estimated effect because students who passed without triggering feedback remain in

their assigned arm. The secondary analysis was also not preregistered and should therefore be treated as exploratory.

Finally, the data come from one introductory Python course at a single U.S. institution, the outcome captures near transfer over a short interval, and the confidence intervals remain broad. The absence of statistical significance does not establish equivalence; modest beneficial or harmful transfer effects remain compatible with the data.

5. CONCLUSION

This secondary analysis of the ProgFeed dataset asked whether assistance that helps students solve a current programming problem leaves them better able to solve a later related problem when the AI is no longer available.

Across three counterbalanced intervention periods, no detectable evidence was found that assignment to LLM-generated natural-language hints or generated failing test cases improved first-attempt performance on the subsequent AI-free pre-lab relative to no AI. This contrasts with the original within-lab finding that natural-language feedback was associated with greater eventual correctness and faster convergence [3].

The combined evidence supports a disciplined interpretation: LLM feedback can be useful for debugging the assisted task, but short-term success should not be assumed to imply transfer. Educational AI should therefore be judged not only by how effectively it helps students finish programs, but also by whether students later demonstrate stronger independent reasoning after the assistance is withdrawn.

ACKNOWLEDGMENT

The authors acknowledge the ML4ED group at the University of Massachusetts Amherst for making the deidentified ProgFeed dataset publicly available. Funding: [Insert funding source and grant number, or state that no external funding was received.] Conflicts of Interest: The author(s) declare no competing interests. Data Availability: The public ProgFeed dataset is available under CC BY 4.0 at <https://github.com/umass-ml4ed/progFeed-dataset-public>.

REFERENCES

- [1] P. Bassner, B. Lenk-Ostendorf, R. Beinstingel, T. Wasner, and S. Krusche, "Less stress, better scores, same learning: The dissociation of performance and learning in AI-supported programming education," *Computers and Education: Artificial Intelligence*, vol. 10, Art. no. 100537, 2026, doi: 10.1016/j.caeai.2025.100537.
- [2] G. Kestin, K. Miller, A. Kiales, T. Milbourne, and G. Ponti, "AI tutoring outperforms in-class active learning: An RCT introducing a novel research-based design in an authentic educational setting," *Scientific Reports*, vol. 15, Art. no. 17458, 2025, doi: 10.1038/s41598-025-97652-6.
- [3] H. Heckal and A. Lan, "A classroom study of LLM-generated feedback intervention in introductory

- programming,” arXiv:2606.08807, 2026, doi: 10.48550/arXiv.2606.08807.
- [4] H. Keuning, J. T. Jeurig, and B. Heeren, “A systematic literature review of automated feedback generation for programming exercises,” *ACM Transactions on Computing Education*, vol. 19, no. 1, Art. no. 3, 2018, doi: 10.1145/3231711.
- [5] J. Hattie and H. Timperley, “The power of feedback,” *Review of Educational Research*, vol. 77, no. 1, pp. 81–112, 2007, doi: 10.3102/003465430298487.
- [6] F. J. Agbo, C. Olivia, G. Oguibe, I. T. Sanusi, and G. Sani, “Computing education using generative artificial intelligence tools: A systematic literature review,” *Computers and Education Open*, vol. 9, Art. no. 100266, 2025, doi: 10.1016/j.caeo.2025.100266.
- [7] P. Manorat, S. Tuarob, and S. Pongpaichet, “Artificial intelligence in computer programming education: A systematic literature review,” *Computers and Education: Artificial Intelligence*, vol. 8, Art. no. 100403, 2025, doi: 10.1016/j.caeai.2025.100403.
- [8] S. Li, J. Liu, and Q. Dong, “Generative artificial intelligence-supported programming education: Effects on learning performance, self-efficacy and processes,” *Australasian Journal of Educational Technology*, vol. 41, no. 3, pp. 1–25, 2025, doi: 10.14742/ajet.9932.
- [9] J. Sweller, J. J. G. van Merriënboer, and F. Paas, “Cognitive architecture and instructional design: 20 years later,” *Educational Psychology Review*, vol. 31, pp. 261–292, 2019, doi: 10.1007/s10648-019-09465-5.
- [10] M. T. H. Chi and R. Wylie, “The ICAP framework: Linking cognitive engagement to active learning outcomes,” *Educational Psychologist*, vol. 49, no. 4, pp. 219–243, 2014, doi: 10.1080/00461520.2014.965823.
- [11] OECD, *OECD Digital Education Outlook 2026: Exploring Effective Uses of Generative AI in Education*. Paris, France: OECD Publishing, 2026, doi: 10.1787/062a7394-en.