A Combined Strategy for the Pedagogical Evaluation of Automated Feedback: Generation, Decision and Fairness*

Badmavasan Kirouchenassamy[1,*], Amel Yessad[1], Sébastien Jolivet[1,2], Sébastien Lallé[1], Matthieu Branthôme[3] and Vanda Luengo[1]

[1]Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
[2]IUFE & TECFA, Université de Genève, Switzerland
[3]University of Rennes, CNRS, IRISA, 22305 Lannion, France

Abstract

Automated feedback is increasingly used in digital learning environments, yet its pedagogical evaluation is too often reduced to a single accuracy or satisfaction metric. We argue that pedagogy is intrinsically multidimensional and that evaluating automated feedback therefore requires a *combined strategy* that addresses, separately and explicitly, (i) the pedagogical validity of the feedback, (ii) the pedagogical alignment of the *decision* of which feedback to deliver and when, and (iii) the *fairness* of the deployed system across learner populations. We describe each layer, illustrate it with our ongoing work in multiple online Python practice platform deployed in French high schools, and outline how the three layers can be operationalised together using a reference pedagogical model, large language models and demographic indicators of equity. Our aim is not to propose a single new metric but to argue for a layered evaluation pipeline that mirrors the multidimensional nature of feedback itself.

Keywords

Pedagogical evaluation, Automated feedback, Programming education,

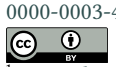
1. Introduction

Feedback is one of the most powerful levers of learning [1, 2], but its effects are neither guaranteed nor uniform, and the same applies to pedagogy more broadly: in the educational-sciences tradition, pedagogy is the deliberate orchestration of instructional actions in relation to a learner, a content and an objective [1, 3], and its quality cannot be evaluated along a single axis. Yet most evaluations of automated feedback today still rely on a narrow set of layers: the accuracy of the message produced by an auto-grader or a large language model (LLM), aggregate *learning gains* from pre/post comparisons, *engagement* signals such as time-on-platform or submission counts, and short student satisfaction surveys [4, 5]. Each is informative in isolation but partial: accuracy says nothing about whether a message respects the institution's pedagogical model; learning gains and engagement aggregate over the feedback strategy itself, so they cannot tell whether the system's per-decision choices were aligned with the intended objective or whether outcomes are distributed equitably across learners. Current practice therefore conflates pedagogical quality with surface correctness plus outcome proxies, leaving the decision and equity dimensions largely invisible. In this position paper, we propose a *combined evaluation strategy* organised along three complementary layers — generation, decision and fairness — illustrated with our ongoing work in two Python programming-practice platforms used in French high schools: AlgoPython[1] and Pyrates[2].

PEAF 2026: Workshop on Pedagogical Evaluation of Automated Feedback, co-located with the Festival of Learning 2026 (AIED, EDM, L@S), June 28, 2026, Seoul, South Korea

*You can use this document as the template for preparing your publication. We recommend using the latest version of the ceurart style.

badmavasan.kirouchenassamy@lip6.fr (B. Kirouchenassamy); amel.yessad@lip6.fr (A. Yessad); sebastien.jolivet@unige.ch (S. Jolivet); sebastien.lalle@lip6.fr (S. Lallé); vanda.luengo@lip6.fr (V. Luengo)
0009-0003-6502-154X (B. Kirouchenassamy); 0000-0001-7575-6433 (A. Yessad); 0000-0003-3915-8465 (S. Jolivet); 0000-0003-4460-8336 (S. Lallé); 0000-0001-8978-0944 (V. Luengo)



[1]www.algopython.fr
[2]www.py-rates.org

Scope. We restrict our attention to *epistemic* feedback: written explanations or hints that scaffold the learner's understanding of a concept. Other forms commonly grouped under the term feedback — failing test cases, "expected vs. observed" diffs, or bare error pointers without elaboration — fall outside this subset. The strategy applies whether the message is picked from a bank of pre-written, expert-authored feedback or generated on the fly by an LLM. Concrete empirical results from our deployed systems will be presented directly at the workshop.

2. Pedagogy is Multidimensional: Two Operational Definitions

Feedback is widely understood as a multidimensional pedagogical act [1, 6, 3]: a feedback episode is the product of at least three pedagogical choices — the *content* of the message, the *moment and form* of its delivery, and the *appropriateness* of that delivery for a given learner — each of which can fail independently. Pedagogically meaningful evaluation must therefore address each separately. To turn this into an actionable strategy we use two definitions throughout.

Pedagogical model. A structured specification, written by domain experts, that defines (i) the elementary *units* of feedback (e.g., a conceptual explanation, a procedural hint, an error pointer, an example); (ii) the *hard boundaries* of each unit — what it must contain and, equally importantly, what it must *not* contain; and (iii) the rules under which units can be *combined* into a feedback episode. A pedagogical model is more than a taxonomy: it is an executable contract that turns a free-form message into a structured object whose validity is decidable.

Pedagogically valid feedback. A message is *pedagogically valid* when it simultaneously satisfies (a) *unit conformity* — it instantiates one or more units within each unit's hard boundaries; (b) *contextual anchoring* — it is tied to the targeted knowledge component (KC), the exercise at hand and the specific error when applicable; and (c) *actionability* — a learner from the target population can understand the message and act on it. Validity in this sense is stronger than surface correctness or fluency: a fluent message that violates a unit's boundary (e.g., a "conceptual explanation" that quietly hands the procedural solution) is invalid even though it reads well.

On this basis we evaluate automated feedback along three complementary layers that mirror the three failure modes: *generation* (Section 3), *decision* (Section 4) and *fairness* (Section 5). A system can pass one and fail another. These are not an exhaustive partition — motivational uptake, metacognitive scaffolding, or longitudinal teacher-in-the-loop integration could be added — but are the dimensions we have so far implemented on a deployed system.

3. Layer 1 – Evaluating Feedback Content Against a Pedagogical Model

The first layer asks whether each produced message is pedagogically valid in the sense above, regardless of whether it was hand-authored, rule-generated or LLM-produced. When feedback is constructed *from* a pedagogical model, evaluation checks that the construction procedure preserves the model's constraints; when it is produced without one, evaluation must verify *post hoc* that each message is consistent with the institution's chosen model. Without an explicit model, "pedagogical quality" has no operational meaning beyond fluency [7, 8].

A reference model for programming feedback. On AlgoPython we use a Reference Praxeological Model (RPM) grounded in the Anthropological Theory of the Didactic [9, 10], which decomposes programming knowledge into *task types*, *techniques* and *logos* (the discourse justifying a technique). A generated message is then a *structured combination of characteristics*, each with explicit boundaries: *logos* (the conceptual unit, no procedural direction); *technical* (a procedural nudge, no working solution); *error_pointed* (explicit identification of a specific error, redirected to the underlying concept); and *with_example_unrelated* / *with_example_related* (an illustrative code example, generic or exercise-specific). A feedback episode is one or more of these characteristics anchored to a single KC, which makes its conformity to the model decidable on a per-characteristic basis.

What this evaluation requires, and why it must be automated. Per item, evaluation checks the three validity conditions of Section 2: unit conformity (e.g., a *logos* message must not contain a working code snippet), anchoring to this KC, exercise and error, and actionability for the target learner — with fluency and, when characteristics co-occur, complementarity rather than redundancy. Performed by hand this is standard didactic-engineering practice (in our initial AlgoPython rollout, 123 items were reviewed by ten instructors [11]); the procedure is feasible there because feedback is produced *offline* — authored once, stored as a pool, selected at runtime — so the per-item review is a one-off cost. But two scale pressures already make purely expert-driven evaluation impractical. The pool design itself becomes intractable as the curriculum grows: covering more concepts at the same depth requires hundreds of additional items. And most contemporary systems do not produce feedback offline at all: LLM-based tutors generate messages live on each interaction, so the population of messages is unbounded and expert review cannot precede delivery. In both cases the per-item, per-dimension evaluation must be at least partly automated.

Automating the evaluation. A central methodological contribution of our strategy is that this review can be automated using LLMs as judges constrained by the pedagogical model and grounded in expert references. In AlgoPython each candidate is scored along five independent dimensions: (i) *characteristic purity* — an LLM judge prompted with the target boundary rules returns a verdict and justification; (ii) *contextual anchoring* — structural checks verify exercise-specific identifiers and consistency with at least one stored valid solution; (iii) *calibration against gold standards* — per characteristic and KC, expert-validated "gold" items define expected length, register and granularity, and semantic-similarity scoring surfaces drift without requiring textual match; (iv) *actionability* — a learner-simulator (a prompted LLM playing a K12 student) reports whether a next step is identifiable; (v) *cross-component coherence* — when multiple characteristics co-occur, they must address complementary facets. The dimensions fail independently (pure but unanchored, anchored but unactionable, calibrated but redundant); they are reported separately and the validity verdict is their conjunction. Each failure surfaces a precise, model-grounded justification, which lets us improve generated feedback dimension by dimension rather than against an opaque "quality" signal.

4. Layer 2 – Evaluating Feedback Decisions Against a Pedagogical Objective

Layer 1 certifies what the system *can* say; Layer 2 asks what it *should* say at a given moment. Whenever multiple instructionally meaningful messages can be delivered, the system must *decide* which one, and that decision is itself a pedagogical act [3, 12]. This holds in both modes: in offline-pool systems the decision is a selector over the curated pool; in LLM tutors it is made either by an explicit strategy prompt or, in its absence, implicitly by the model itself. The decision is therefore always present, and "why was *this* feedback given to *this* learner at *this* moment?" is a legitimate question regardless of architecture.

What we evaluate. The decision layer is evaluated by measuring the *impact* of the system's decisions against an explicitly declared *instructional objective*. The objective makes the evaluation pedagogical rather than operational: a *learning-gains* objective calls for durable-mastery measures (PFA-based skill estimates [13], post-test gains, reduced error recurrence), whereas *fast problem-solving* calls for completion rate and interaction count. Different objectives can yield different verdicts on the same logs, so the objective must be declared up front. Concretely, this layer needs a declared objective, a measurement of decision impact against it on real data, and a comparison to plausible alternative strategies.

A working case. The initial AlgoPython deployment used a randomised rule-based selector over the validated pool of feedback items. Analysis of 25,000+ logged submission–feedback interactions showed that, without an explicit objective, the selector *did* deliver feedback at every step but was not consistently useful — operational without being instructionally effective. We therefore trained offline decision policies that align selection with an explicit objective by formulating it as a Markov

Decision Process whose reward function encodes that objective [11]. Off-policy evaluation [14, 15] under five reward functions (rapid completion to longer-term mastery gain) showed that the optimal feedback distribution differs systematically across objectives — no single distribution serves all goals. For example, completion-oriented rewards push the policy toward *technical* and *error_pointed* feedback (procedural nudges that move the learner past the current bug), whereas learning-oriented rewards shift probability mass toward *logos* and *with_example_related* feedback (conceptual scaffolding and worked examples that consolidate the underlying skill). The methodological point is that an explicit pedagogical objective necessary: one can ask whether the system's behaviour is aligned with the stated objective, a question that has no answer when the objective is only implicit, which is the case in most automated feedback systems in recent times that leverage LLMs. However, once a clear pedagogical objective has been defined, we can additionally draw on various data sources–such as log data and pre-test/post-test results–to determine whether the system is aligned with that objective or not.

5. Layer 3 – Evaluating Fairness Across Learner Populations

Layers 1 and 2 concern *what* the system produces and *how* it decides; the third concerns *for whom* it works. A policy that is valid at the item level and well-aligned with an objective on average can still under-serve specific learner groups — a pedagogical failure as much as a content or decision failure. Algorithmic fairness in education has documented such disparities in predictive models [16, 17]; the same scrutiny must extend to feedback, which intervenes on learners more directly than prediction. Algopython and Pyrates is deployed across a network of French high schools, which gives us access to school-level indicators published by the French Ministry of Education [3]: the *Indice de Position Sociale* (IPS, socio-economic background), the *taux de réussite* (baccalauréat success rate) and the *taux de mention* (honours rate). We use these to partition the user base and evaluate the system's pedagogical outcomes — task completion, error recurrence and PFA-based skill mastery [13] — on each group separately, asking whether the same policy yields comparable mastery gains across IPS groups, whether it widens or narrows error-recurrence gaps, and whether differences in feedback distribution across school profiles are instructionally justified or simply an artefact of the deployed policy. Treating fairness as a distinct layer has a further benefit: subgroup imbalances surfaced here can feed back into Layer 2 as additional constraints or auxiliary reward terms, so that fairness evaluation drives policy revision rather than acting as a passive audit. These demographic indicators are one grouping basis among several: in parallel work we are investigating data-driven groupings derived directly from student interaction logs, with the aim of comparing the pedagogical effectiveness of the same feedback policy across learner groups identified without sensitive attributes.

6. Conclusion

We have argued that the pedagogical evaluation of automated feedback cannot be reduced to a single metric, because pedagogy itself is multidimensional. From operational definitions of *pedagogical model* and *pedagogically valid feedback*, we proposed a combined strategy in three complementary layers — *generation*, *decision* and *fairness* (Sections 3–5) — each illustrated by a working case on a deployed Python platform. A system that fails any one layer is pedagogically deficient even if it scores well on the others. Several limitations frame our future work: LLM-judge reliability is under study against expert annotation at scale, and gold-anchor calibration is sensitive to reference-set coverage; scalar rewards are a strong simplification, and offline RL is bounded by the logging policy's coverage; school-level indicators like IPS are coarse, and learner-level fairness is methodologically hard when sensitive attributes are not collected; and our validation is restricted to one discipline, leaving cross-domain generalisation and the closing of the three layers into a continuous loop as the central open questions we hope to raise at PEAF 2026.

[3] https://data.education.gouv.fr/pages/recherche-etablissement/

References

[1] J. Hattie, H. Timperley, The power of feedback, Review of Educational Research 77 (2007) 81–112.

[2] B. Wisniewski, K. Zierer, J. Hattie, The power of feedback revisited: A meta-analysis of educational feedback research, Frontiers in Psychology 10 (2020) 3087.

[3] S. Narciss, S. Sosnovsky, L. Schnaubert, E. Andrès, A. Eichelmann, G. Goguadze, E. Melis, Exploring feedback and student characteristics relevant for personalizing feedback strategies, Computers & Education 71 (2014) 56–76.

[4] A. P. Cavalcanti, A. Barbosa, R. Carvalho, F. Freitas, Y.-S. Tsai, D. Gašević, R. F. Mello, Automatic feedback in online learning environments: A systematic literature review, Computers and Education: Artificial Intelligence 2 (2021) 100027.

[5] M. Messer, N. C. C. Brown, M. Kölling, M. Shi, Automated grading and feedback tools for programming education: A systematic review, ACM Transactions on Computing Education (2024).

[6] V. J. Shute, Focus on formative feedback, Review of Educational Research 78 (2008) 153–189.

[7] N. Scholz, M. H. Nguyen, A. Singla, T. Nagashima, Partnering with AI: A pedagogical feedback system for LLM integration into programming education, in: European Conference on Technology Enhanced Learning, Springer, 2025, pp. 243–248.

[8] M. Liffiton, B. E. Sheese, J. Savelka, P. Denny, CodeHelp: Using large language models with guardrails for scalable support in programming classes, in: Proceedings of the 23rd Koli Calling International Conference on Computing Education Research, 2023, pp. 1–11.

[9] S. Jolivet, Towards a praxeological reference model for programming: Construction and presentation, in: Pre-proceedings of the 8th International Conference on the Anthropological Theory of the Didactic, Barcelona, Spain, 2026, pp. 139–151.

[10] S. Jolivet, B. Kirouchenassamy, A. Yessad, V. Luengo, Modélisation et décision de rétroactions épistémiques dans l'environnement AlgoPython, in: Actes de la 12e conférence sur les Environnements Informatiques pour l'Apprentissage Humain (EIAH25), 2025, pp. 172–190.

[11] B. Kirouchenassamy, A. Yessad, S. Jolivet, V. Luengo, Offline reinforcement learning for adaptive feedback in online programming education, 2026. Accepted at AIED 2026.

[12] F. Gutierrez, J. Atkinson, Adaptive feedback selection for intelligent tutoring systems, Expert Systems with Applications 38 (2011) 6146–6152.

[13] P. I. Pavlik Jr., H. Cen, K. R. Koedinger, Performance Factors Analysis – A New Alternative to Knowledge Tracing, Technical Report, Online submission, 2009.

[14] A. R. Mahmood, H. P. van Hasselt, R. S. Sutton, Weighted importance sampling for off-policy learning with linear function approximation, in: Advances in Neural Information Processing Systems, volume 27, 2014.

[15] B. Hao, X. Ji, Y. Duan, H. Lu, C. Szepesvári, M. Wang, Bootstrapping fitted Q-evaluation for off-policy inference, in: International Conference on Machine Learning (ICML), PMLR, 2021, pp. 4074–4084.

[16] R. S. Baker, A. Hawn, Algorithmic bias in education, International Journal of Artificial Intelligence in Education 32 (2022) 1052–1092.

[17] R. F. Kizilcec, H. Lee, Algorithmic fairness in education, in: The Ethics of Artificial Intelligence in Education, Routledge, 2022, pp. 174–202.